Online Multiple Instance Learning Applied to Hand Detection in a Humanoid Robot

Carlo Ciliberto
Robotics, Brain and Cognitive Sciences, IIT
Machine Learning Day
June 8th, 2010

Overview

- Why the hand?
- Multiple Instance Learning
- Online Boosting
- Online Multiple Instance Learning
- Experiments
- Conclusions

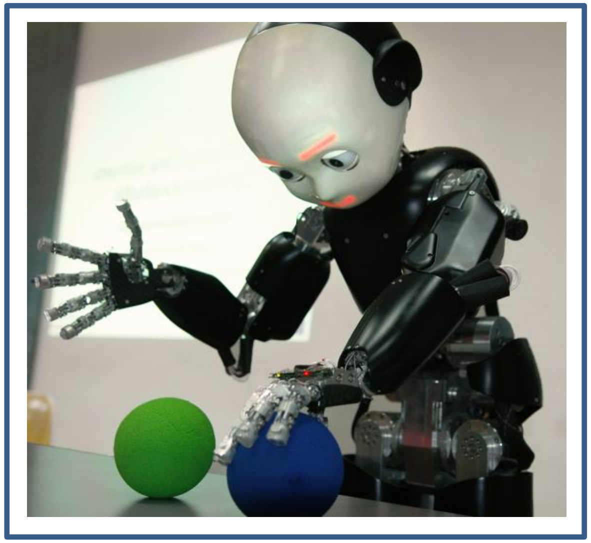
Overview

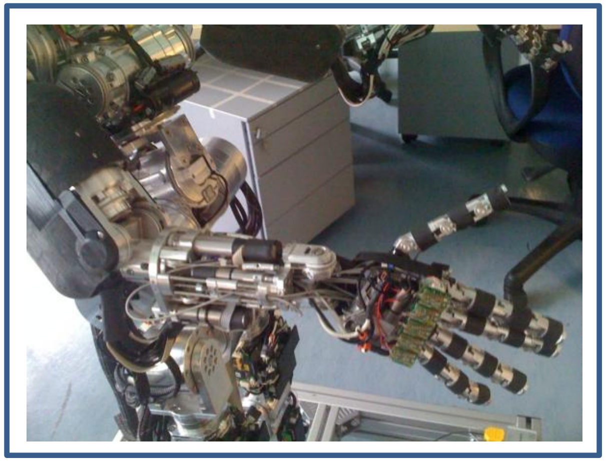
- Why the hand?
- Multiple Instance Learning
- Online Boosting
- Online Multiple Instance Learning
- Experiments
- Conclusions

Necessary to Exploration

Exploration is the best way for an autonomous agent to obtain information regarding the surrounding environment.

In order to interact with the environment, a robotic system needs to recognize the means through which, such interaction is happening.



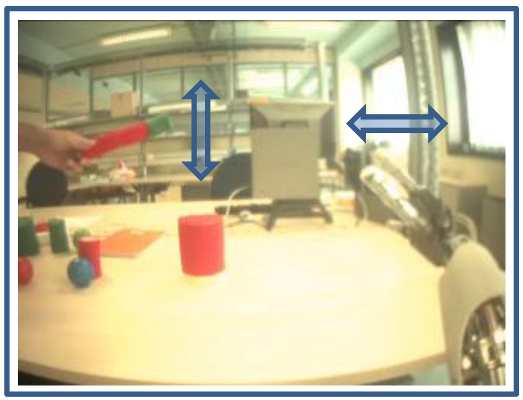


For a humanoid robot this is the case of its **hands**.

Unique Object

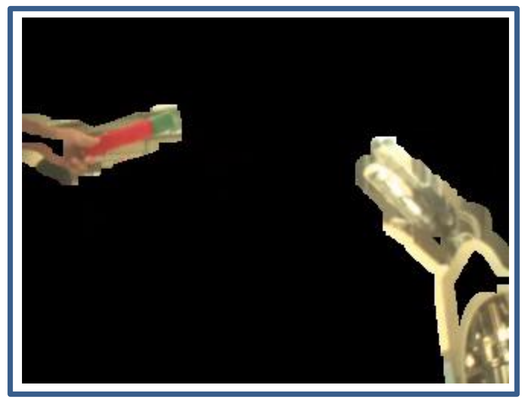
The hand is a very unique object to learn. The poses in which it appears to the robot are strongly connected to the current **joint** and **motor states.**

Visual motion detection and **motor** information can be combined to roughly infer whether the robot's hand is present in the Field of View.



Motors state: ON

$\oplus$



Motion Detected

$=$

Hand is probably in the FoV

Even if inaccurate, this expedient can be exploited to apply supervised learning techniques in a less supervised way

Online Learning

Changes in an unstructured environment are unpredictable.

In order to keep a robust representation of the world, an autonomous agent must be able to **integrate** previous knowledge with new data as it arrives.

We are after an online framework able to constantly keep track of the robotic hand’s structure even when it is subject to changes in appearance. (e.g. illumination, rotation).

Overview

- Why the hand?
- Multiple Instance Learning
- Online Boosting
- Online Multiple Instance Learning
- Experiments
- Conclusions

Multiple Instance Learning

Multiple Instance Learning

- In the MIL framework, examples are seen as **bags of instances**.

Multiple Instance Learning

- In the MIL framework, examples are seen as **bags of instances**.
- A bag is positive if it contains at least one positive instance.

Multiple Instance Learning

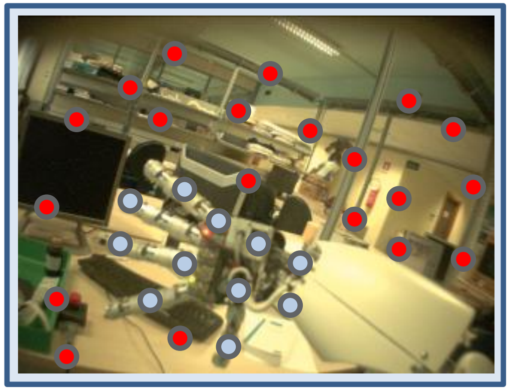
- In the MIL framework, examples are seen as **bags of instances**.
- A bag is positive if it contains at least one positive instance.
- The **individual label** of an instance is not known. Only the label of the entire bag.

Multiple Instance Learning

- In the MIL framework, examples are seen as **bags of instances**.
- A bag is positive if it contains at least one positive instance
- The **individual label** of an instance is not known. Only the label of the entire bag

Multiple Instance Learning

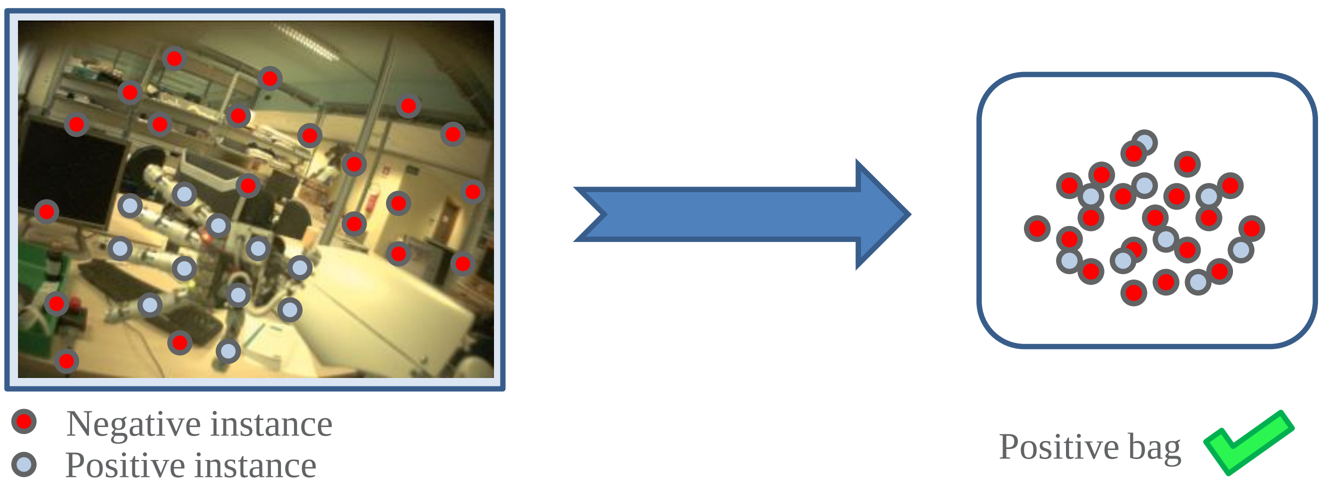
- In the MIL framework, examples are seen as **bags of instances**.
- A bag is positive if it contains at least one positive instance
- The **individual label** of an instance is not known. Only the label of the entire bag



Negative instance
Positive instance

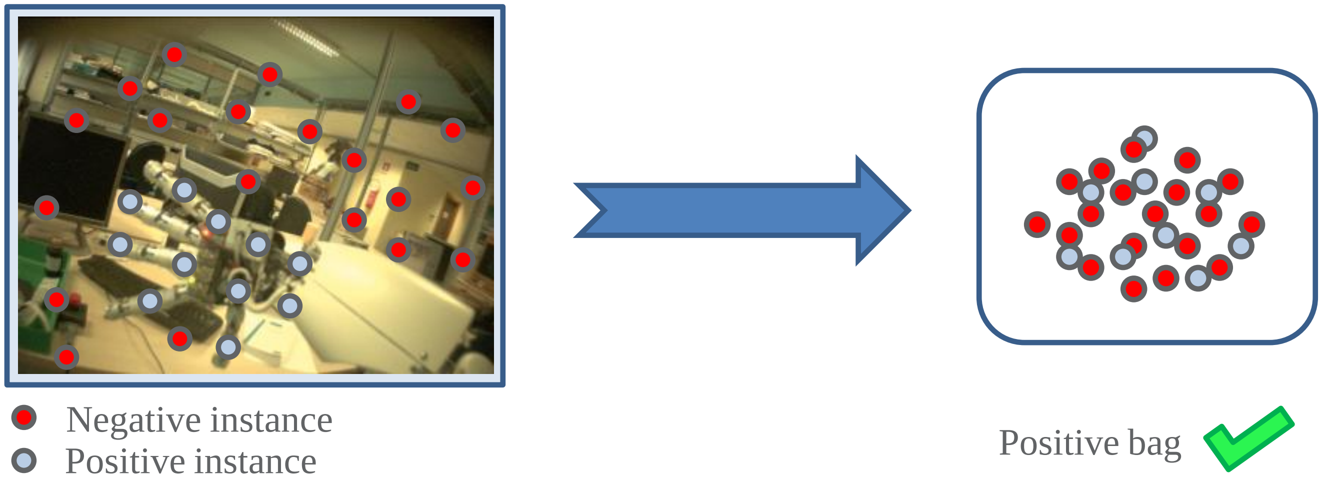
Multiple Instance Learning

- In the MIL framework, examples are seen as **bags of instances**.
- A bag is positive if it contains at least one positive instance
- The **individual label** of an instance is not known. Only the label of the entire bag



Multiple Instance Learning

- In the MIL framework, examples are seen as **bags of instances**.
- A bag is positive if it contains at least one positive instance
- The **individual label** of an instance is not known. Only the label of the entire bag



Goal: Correctly classify the bags without knowing exactly which instances were responsible for the positive/negative label of the bag.

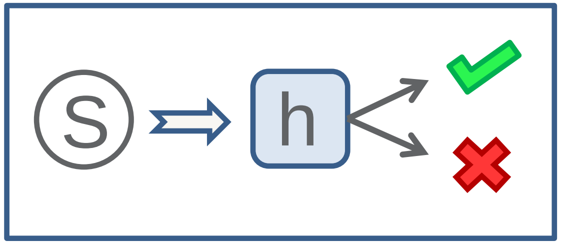
MIL & Boosting

MIL & Boosting

Boosting is a class of techniques employed in data classification.

MIL & Boosting

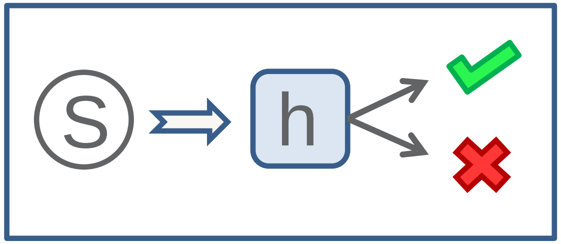
Boosting is a class of techniques employed in data classification.



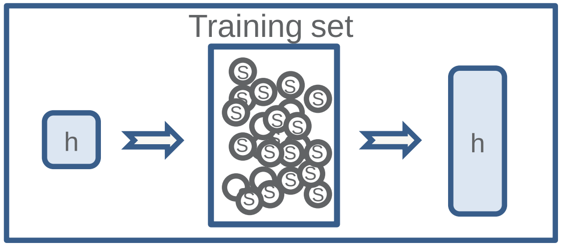
Weak Learner: binary function that classifies data samples as positive or negative.

MIL & Boosting

Boosting is a class of techniques employed in data classification.



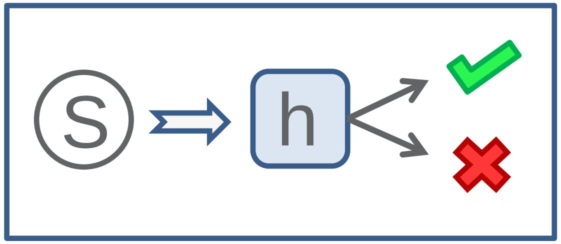
Weak Learner: binary function that classifies data samples as positive or negative.



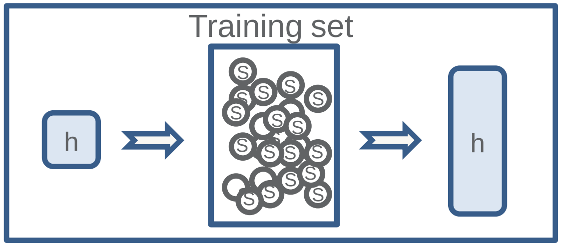
Weights are associated to weak learners according to their accuracy over a given training set.

MIL & Boosting

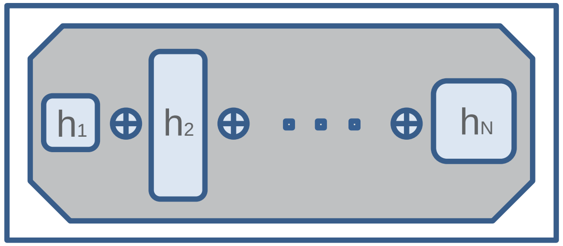
Boosting is a class of techniques employed in data classification.



Weak Learner: binary function that classifies data samples as positive or negative.



Weights are associated to weak learners according to their accuracy over a given training set.



Weak learners are combined to obtain an accurate **strong classifier.**

MIL & Boosting: AdaBoost

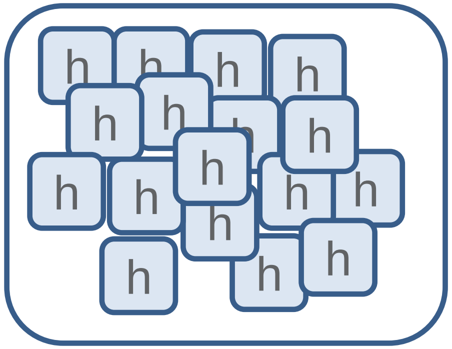
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.

MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.

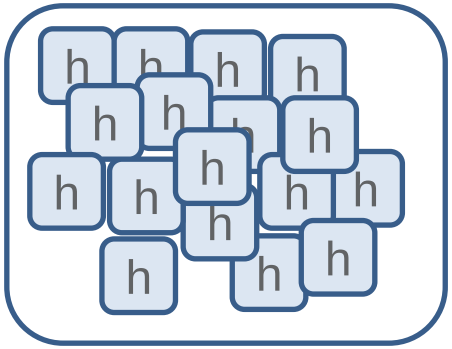
Weak learners



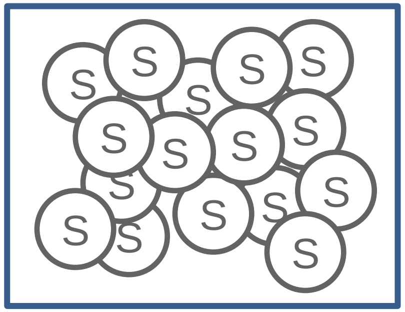
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.

Weak learners

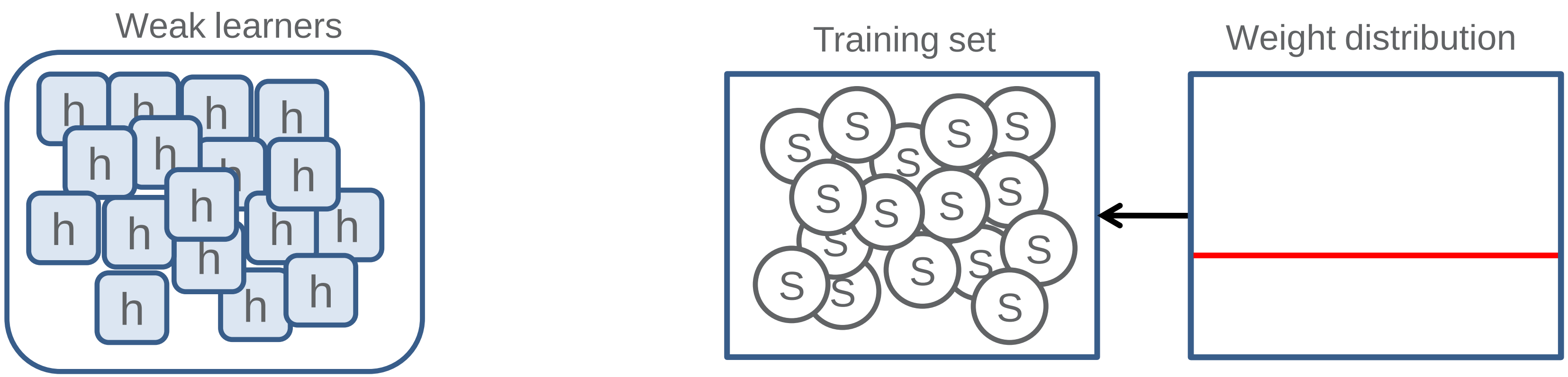


Training set



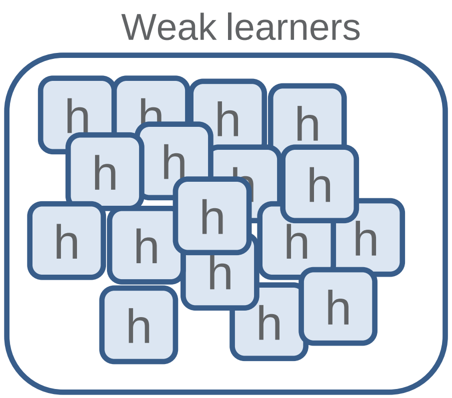
MIL & Boosting: AdaBoost

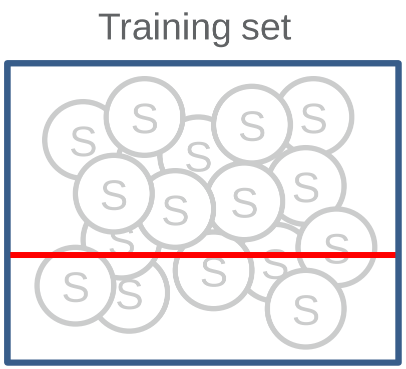
AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



MIL & Boosting: AdaBoost

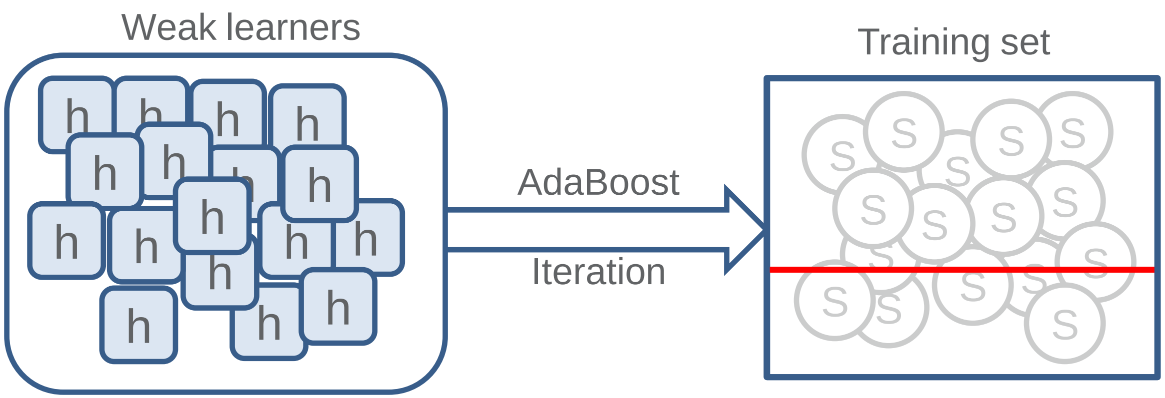
AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.





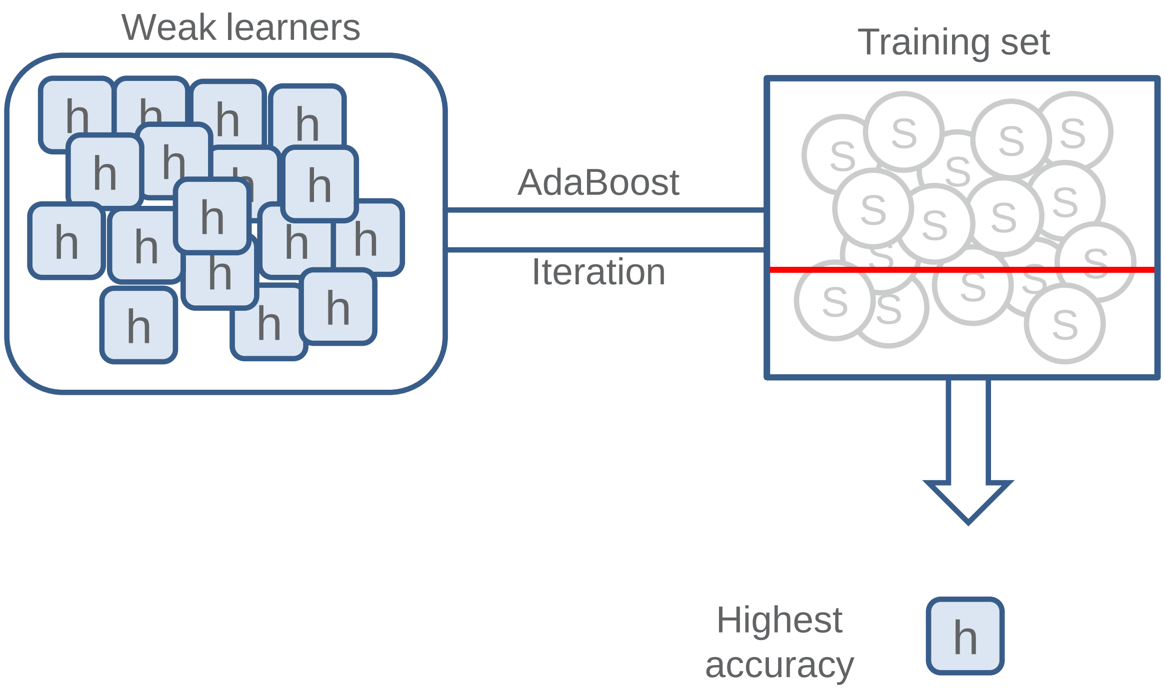
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



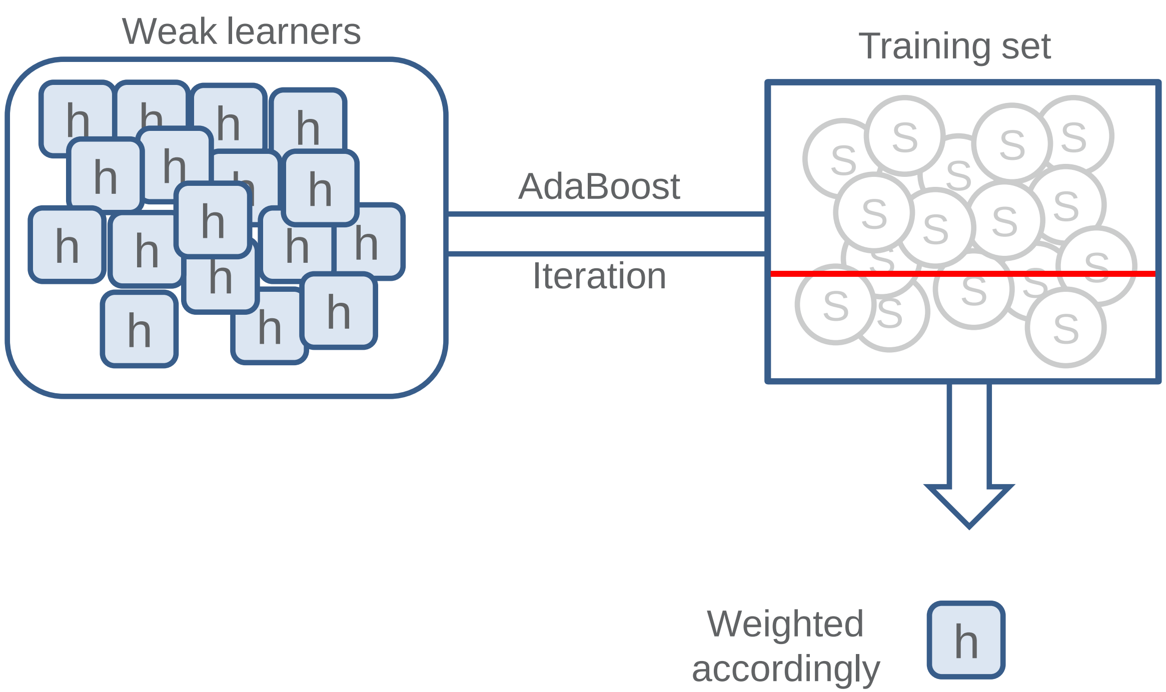
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



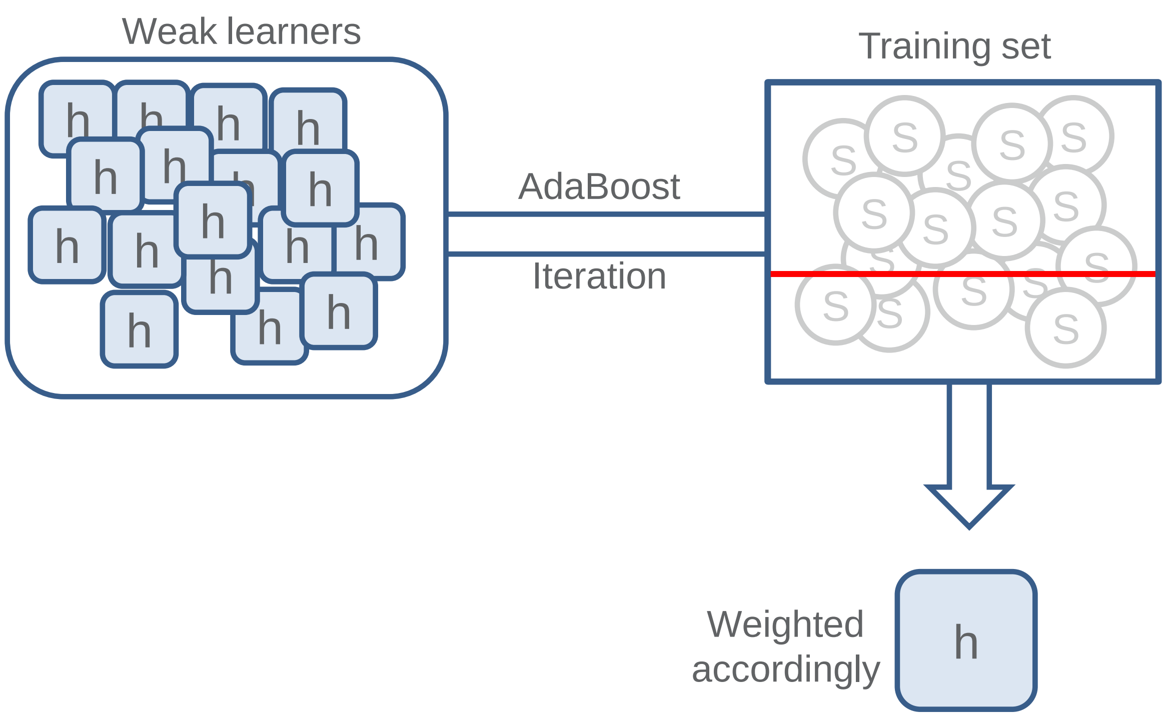
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



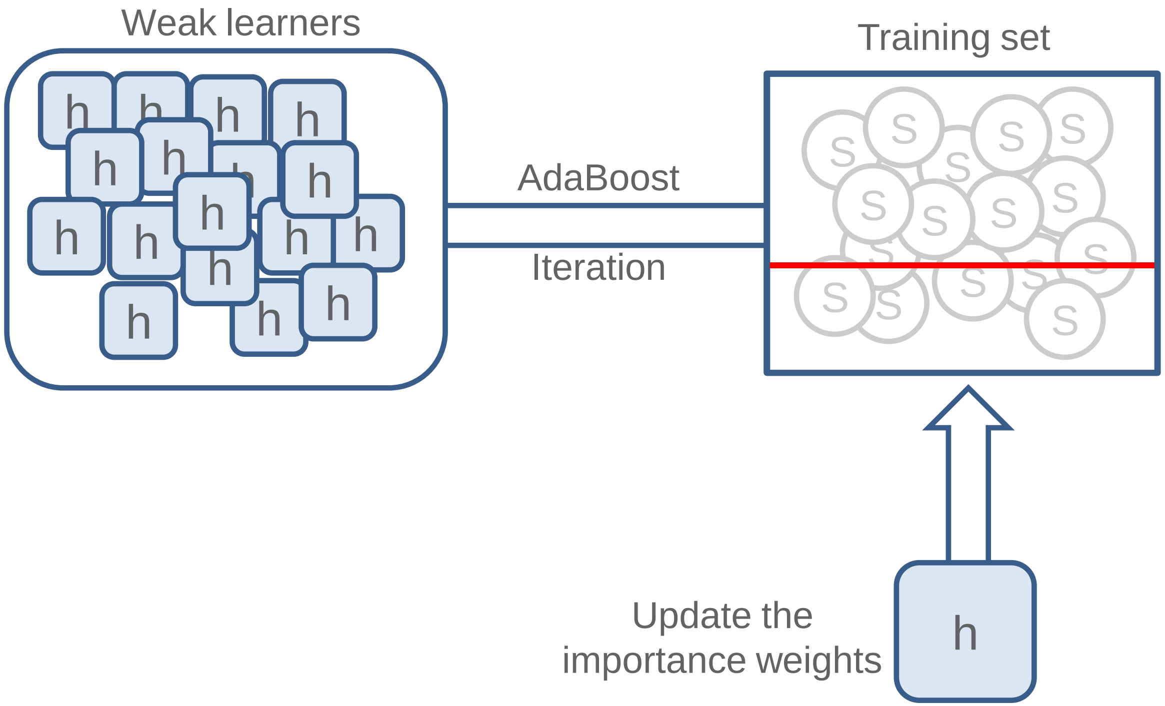
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



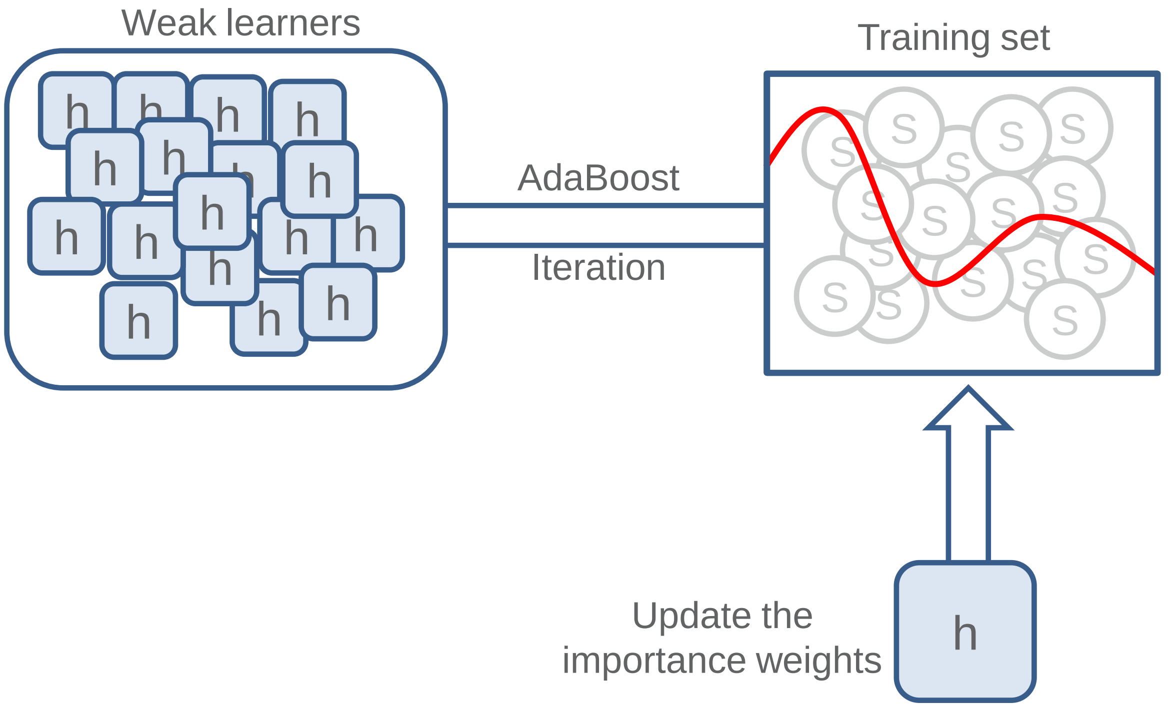
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



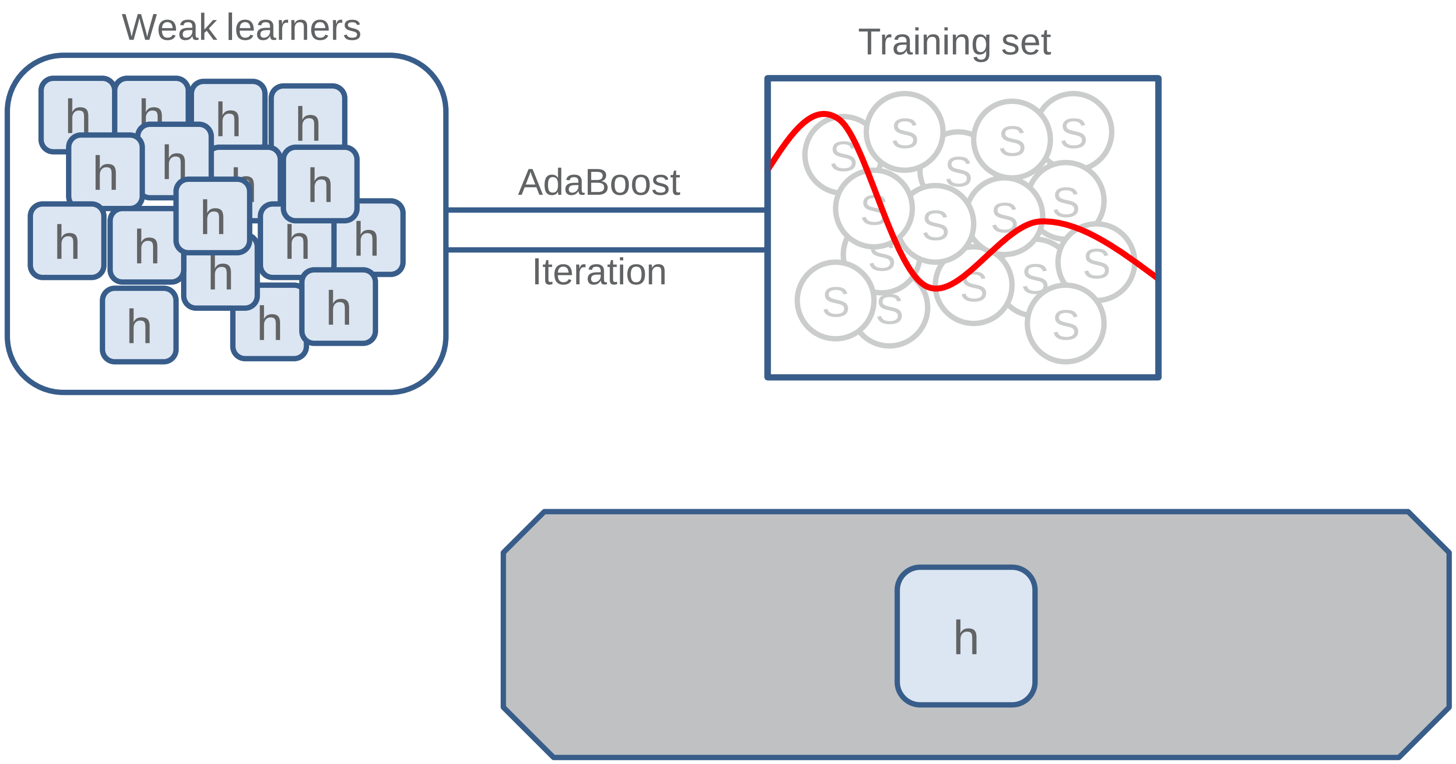
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



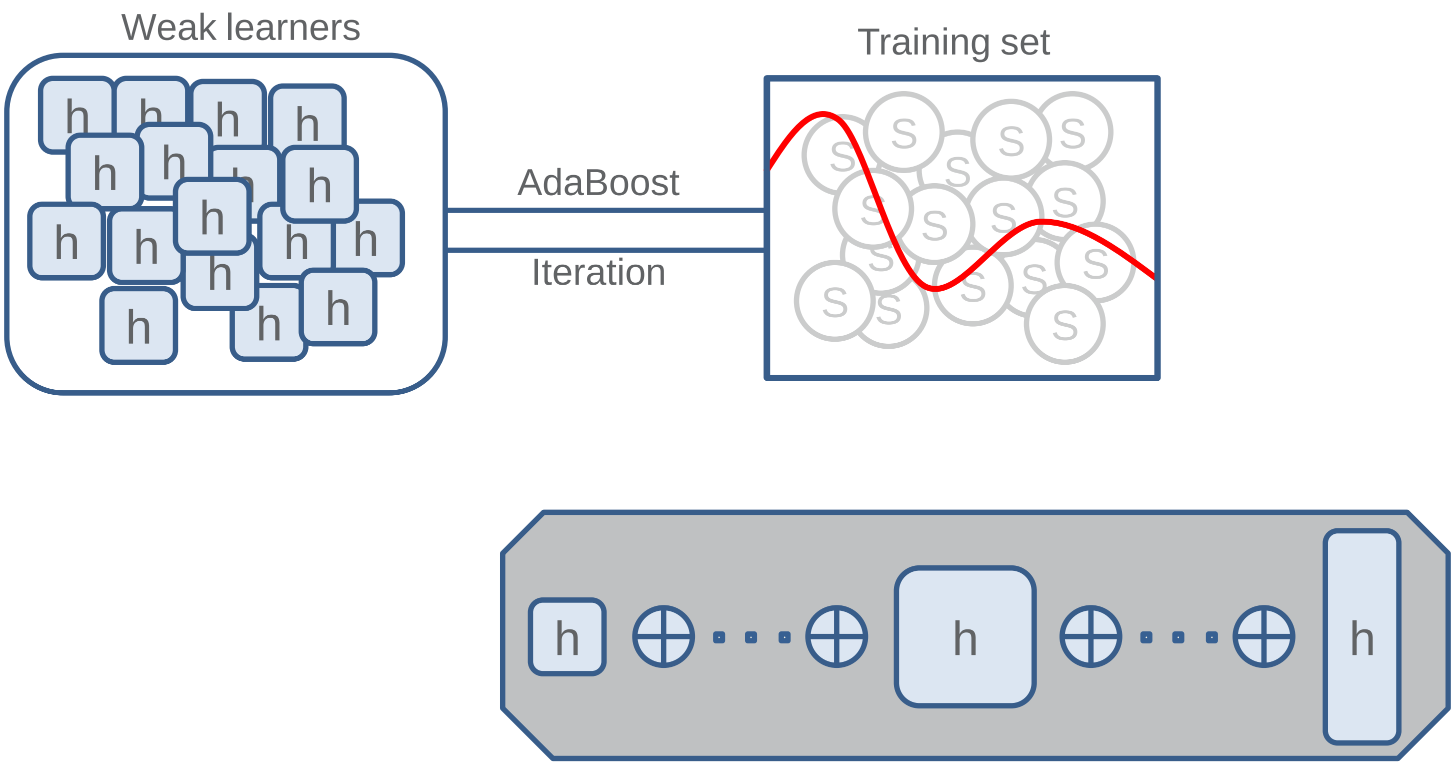
MIL & Boosting: AdaBoost

AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



MIL & Boosting: AdaBoost

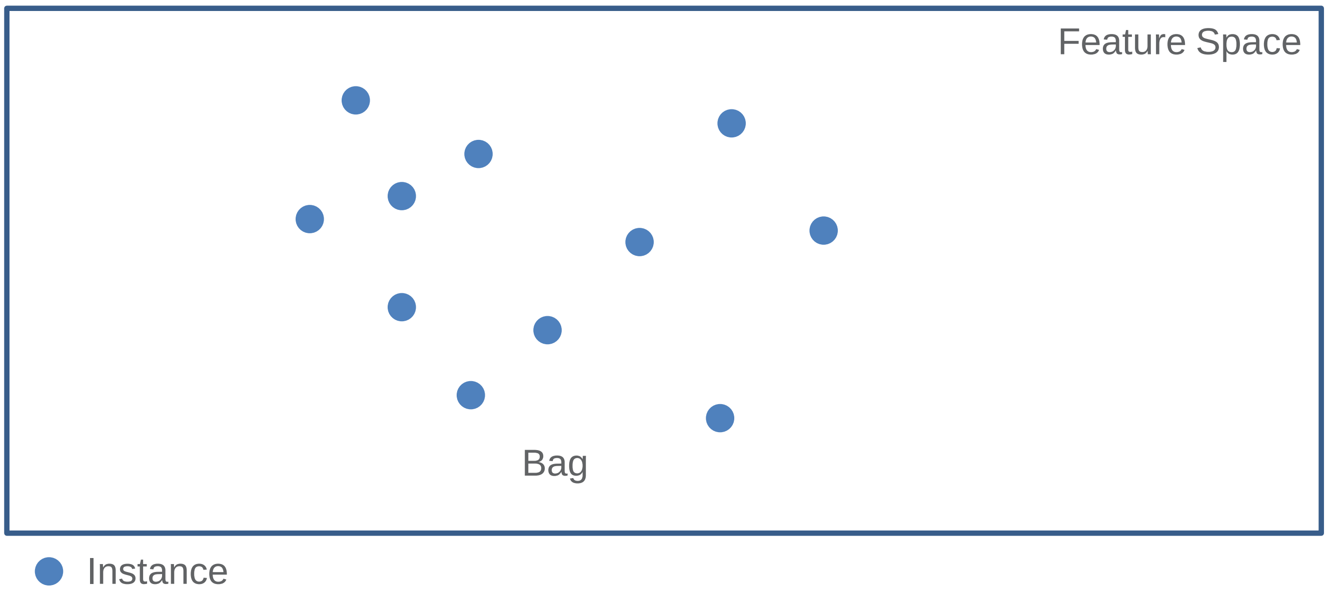
AdaBoost adopts an iterative greedy strategy to select which weak learners combine in the strong classifier.



MIL & Boosting: MIL balls

MIL & Boosting: MIL balls

A bag is a collection of instances in the feature space.

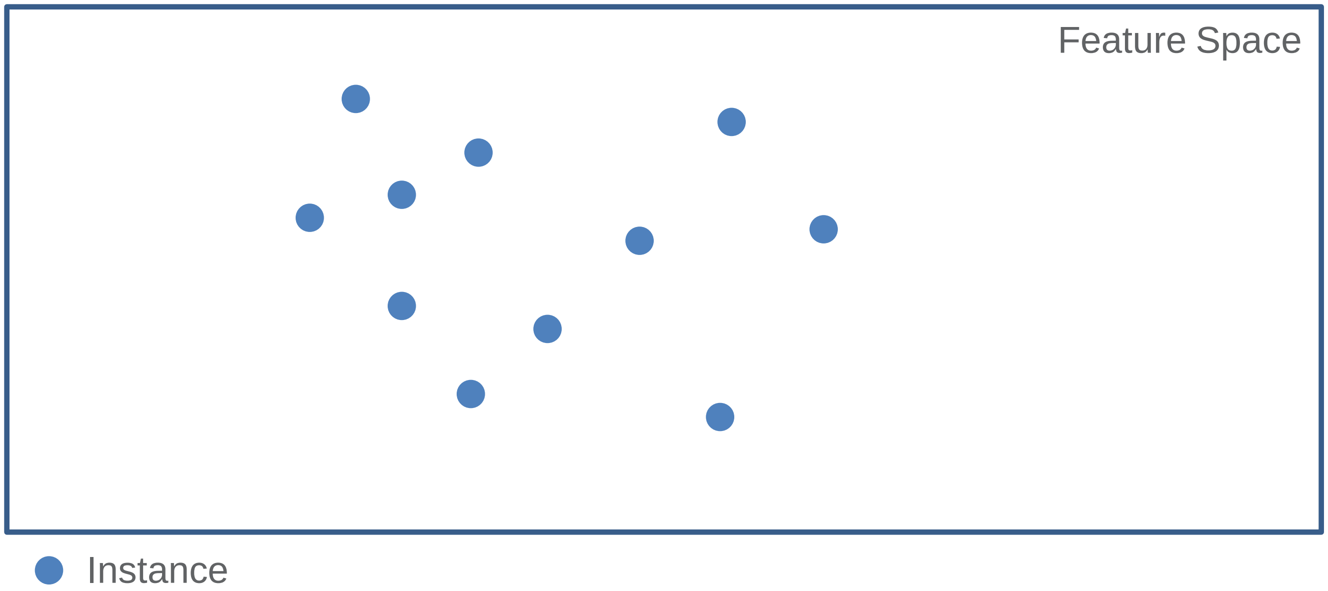


Auer et al. 2004

MIL & Boosting: MIL balls

A bag is a collection of instances in the feature space.

We define a MIL weak learner as a ball centered on a point in the feature space.

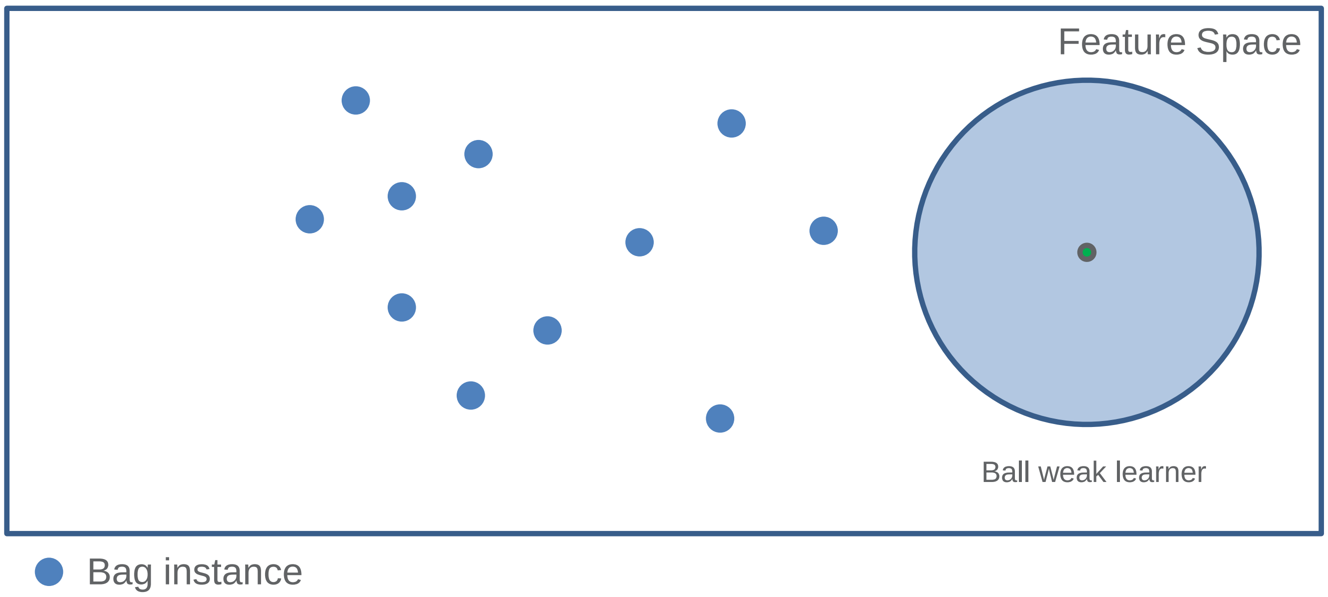


Auer et al. 2004

MIL & Boosting: MIL balls

A bag is a collection of instances in the feature space.

We define a MIL weak learner as a ball centered on a point in the feature space.



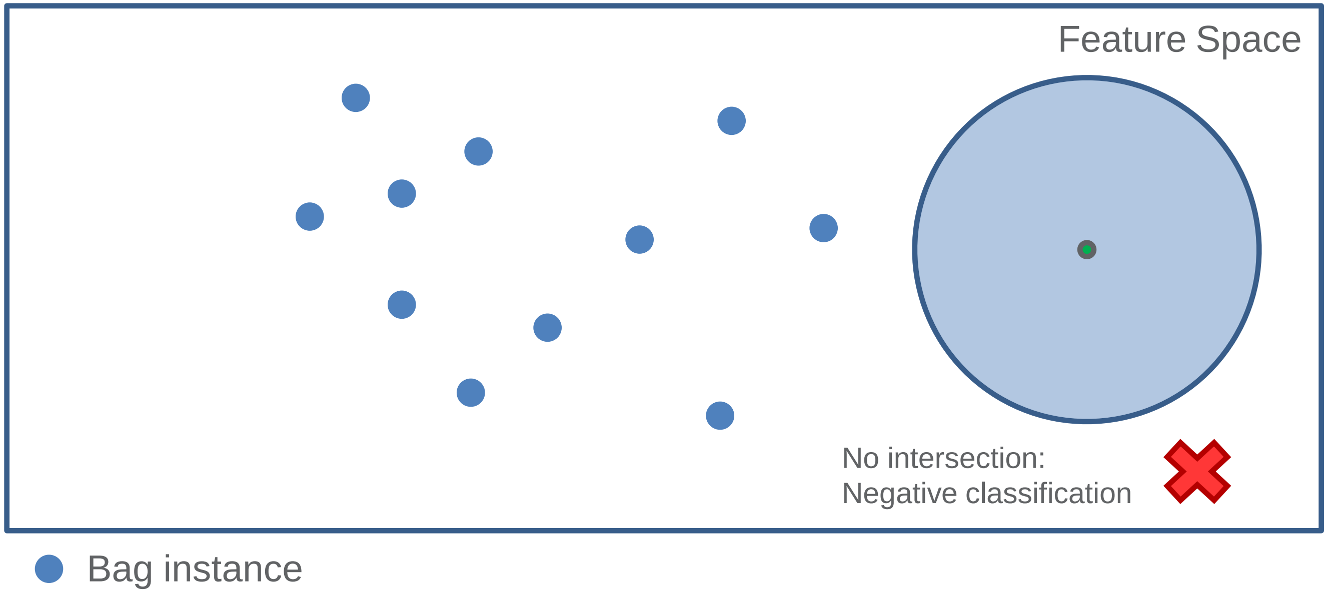
Auer et al. 2004

MIL & Boosting: MIL balls

A bag is a collection of instances in the feature space.

We define a MIL weak learner as a ball centered on a point in the feature space.

A ball classifies positively only the bags that it intersects.



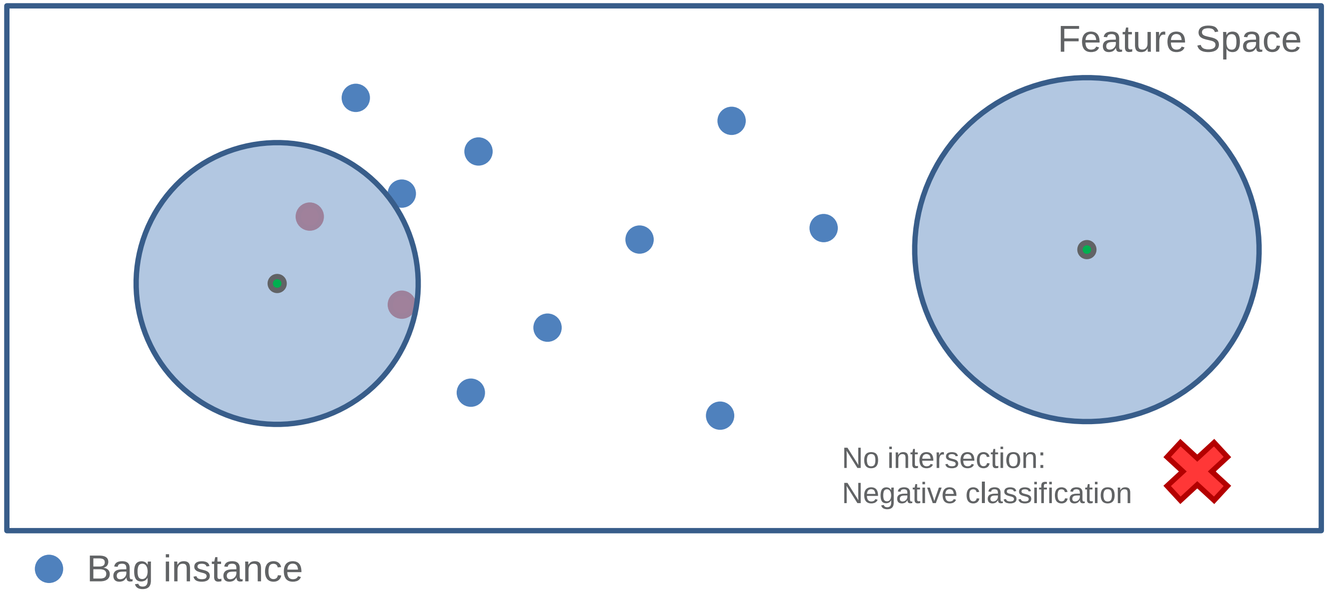
Auer et al. 2004

MIL & Boosting: MIL balls

A bag is a collection of instances in the feature space.

We define a MIL weak learner as a ball centered on a point in the feature space.

A ball classifies positively only the bags that it intersects.



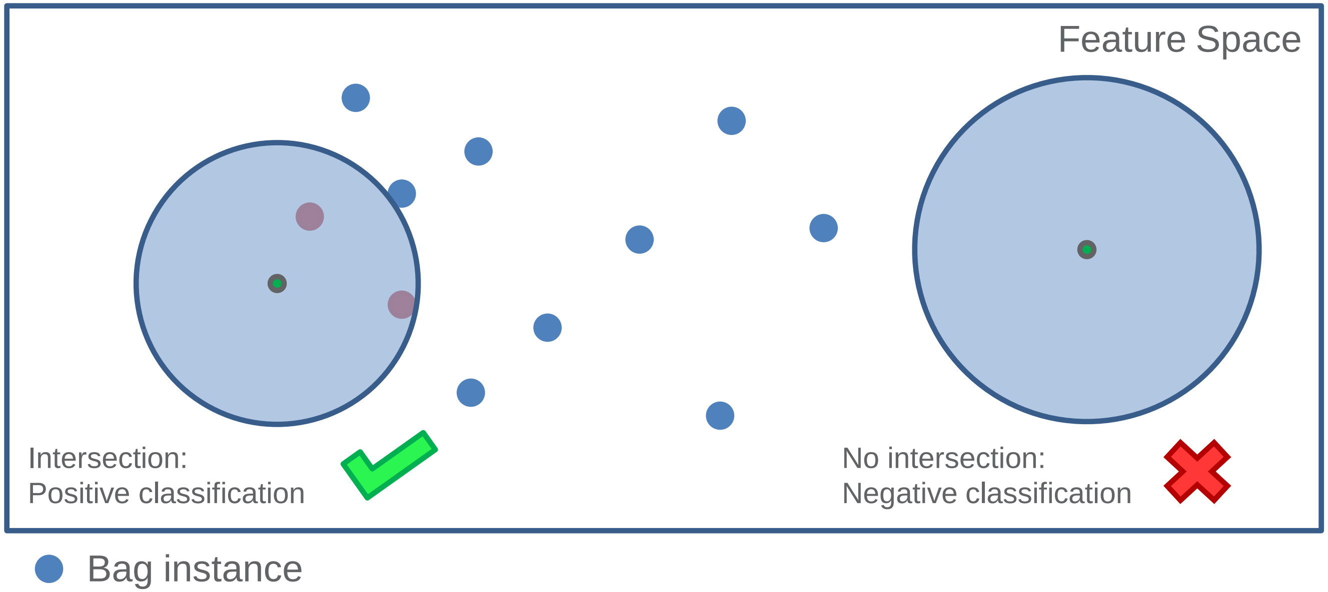
Auer et al. 2004

MIL & Boosting: MIL balls

A bag is a collection of instances in the feature space.

We define a MIL weak learner as a ball centered on a point in the feature space.

A ball classifies positively only the bags that it intersects.



Auer et al. 2004

Overview

- Why the hand?
- Multiple Instance Learning
- Online Boosting
- Online Multiple Instance Learning
- Experiments
- Conclusions

Online Boosting

Online Boosting

Initialization.

- Let $\mathcal{H} = \{h_1, ..., h_N\}$ be a set of weak classifiers with Learning Principle L.
- Set $\lambda_n^w = \lambda_n^c = 0 \;\; \forall n \in \{1,...,N\}$.

Training.

At each iteration step t a novel sample I_t is presented to the system:

- Set the importance weight of the sample to $\lambda = 1$.
- For $n \in \{1,...,N\}$ do:
  1. update $h_n \leftarrow L(h_n, I_t, \lambda)$
  2. if h_n correctly classifies I_t:
     - $\lambda_n^c \leftarrow \lambda_n^c + \lambda$; $\epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}$; $\lambda \leftarrow \lambda \frac{1}{2(1-\epsilon_n)}$

     else:
     - $\lambda_n^w \leftarrow \lambda_n^w + \lambda$; $\epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}$; $\lambda \leftarrow \lambda \frac{1}{2\epsilon_n}$
  3. Define the relevance weight of the n-th weak learner as $\alpha_n = \log\left(\frac{1-\epsilon_n}{\epsilon_n}\right)$
- end.

Strong Classifier.

After every learning iteration, the score assigned by the strong classifier to a bag $I \in \mathcal{I}$ is:

$$S(I) = \sum_{n=1}^{N} \alpha_n \cdot h_n(I) \; .$$

Oza et al. 2001

Online Boosting

Initialization.

- Let $\mathcal{H} = \{h_1, ..., h_N\}$ be a set of weak classifiers with Learning Principle L.
- Set $\lambda_n^w = \lambda_n^c = 0 \ \ \forall n \in \{1,...,N\}$.

Training.

At each iteration step t a novel sample I_t is presented to the system:

- Set the importance weight of the sample to $\lambda = 1$.
- For $n \in \{1,...,N\}$ do:
  1. update $h_n \leftarrow L(h_n, I_t, \lambda)$
  2. if h_n correctly classifies I_t:
     - $\lambda_n^c \leftarrow \lambda_n^c + \lambda$; $\epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}$; $\lambda \leftarrow \lambda \frac{1}{2(1-\epsilon_n)}$

     else:
     - $\lambda_n^w \leftarrow \lambda_n^w + \lambda$; $\epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}$; $\lambda \leftarrow \lambda \frac{1}{2\epsilon_n}$
  3. Define the relevance weight of the n-th weak learner as $\alpha_n = \log\left(\frac{1-\epsilon_n}{\epsilon_n}\right)$
- end.

Strong Classifier.

After every learning iteration, the score assigned by the strong classifier to a bag $I \in \mathcal{I}$ is:

$$S(I) = \sum_{n=1}^{N} \alpha_n \cdot h_n(I) \, .$$

Oza et al. 2001

Online Boosting

Initialization.

- Let $\mathcal{H} = \{h_1 ,..., h_N\}$ be a set of weak classifiers with Learning Principle L.
- Set $\lambda_n^w = \lambda_n^c = 0 \;\; \forall n \in \{1,...,N\}$.

Training.

At each iteration step t a novel sample I_t is presented to the system:

- Set the importance weight of the sample to $\lambda = 1$.
- For $n \in \{1,...,N\}$ do:
  1. update $h_n \leftarrow L(h_n, I_t, \lambda)$
  2. if h_n correctly classifies I_t:
     - $\lambda_n^c \leftarrow \lambda_n^c + \lambda\,; \;\; \epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}\,; \;\; \lambda \leftarrow \lambda \frac{1}{2(1-\epsilon_n)}$

     else:
     - $\lambda_n^w \leftarrow \lambda_n^w + \lambda\,; \;\; \epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}\,; \;\; \lambda \leftarrow \lambda \frac{1}{2\epsilon_n}$
  3. Define the relevance weight of the n-th weak learner as $\alpha_n = \log\left(\frac{1-\epsilon_n}{\epsilon_n}\right)$
- end.

Strong Classifier.

After every learning iteration, the score assigned by the strong classifier to a bag $I \in \mathcal{I}$ is:

$$S(I) = \sum_{n=1}^{N} \alpha_n \cdot h_n(I)\,.$$

Oza et al. 2001

Online Boosting

Initialization.

- Let $\mathcal{H} = \{h_1, ..., h_N\}$ be a set of weak classifiers with Learning Principle L.
- Set $\lambda_n^w = \lambda_n^c = 0 \;\; \forall n \in \{1,...,N\}$.

Training.

At each iteration step t a novel sample I_t is presented to the system:

- Set the importance weight of the sample to $\lambda = 1$.
- For $n \in \{1,...,N\}$ do:
  1. update $h_n \leftarrow L(h_n, I_t, \lambda)$
  2. if h_n correctly classifies I_t:
     - $\lambda_n^c \leftarrow \lambda_n^c + \lambda \,;\;\; \epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w} \,;\;\; \lambda \leftarrow \lambda \frac{1}{2(1-\epsilon_n)}$

     else:
     - $\lambda_n^w \leftarrow \lambda_n^w + \lambda \,;\;\; \epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w} \,;\;\; \lambda \leftarrow \lambda \frac{1}{2\epsilon_n}$
  3. Define the relevance weight of the n-th weak learner as $\alpha_n = \log\left(\frac{1-\epsilon_n}{\epsilon_n}\right)$
- end.

Strong Classifier.

After every learning iteration, the score assigned by the strong classifier to a bag $I \in \mathcal{I}$ is:

$$S(I) = \sum_{n=1}^{N} \alpha_n \cdot h_n(I) \,.$$

Oza et al. 2001

Online Boosting

Initialization.

- Let $\mathcal{H} = \{h_1, ..., h_N\}$ be a set of weak classifiers with Learning Principle L.
- Set $\lambda_n^w = \lambda_n^c = 0 \ \forall n \in \{1,...,N\}$.

Training.

At each iteration step t a novel sample I_t is presented to the system:

- Set the importance weight of the sample to $\lambda = 1$.
- For $n \in \{1,...,N\}$ do:
  1. update $h_n \leftarrow L(h_n, I_t, \lambda)$
  2. if h_n correctly classifies I_t:
     - $\lambda_n^c \leftarrow \lambda_n^c + \lambda$; $\epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}$; $\lambda \leftarrow \lambda \frac{1}{2(1-\epsilon_n)}$

     else:
     - $\lambda_n^w \leftarrow \lambda_n^w + \lambda$; $\epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}$; $\lambda \leftarrow \lambda \frac{1}{2\epsilon_n}$
  3. Define the relevance weight of the n-th weak learner as $\alpha_n = \log\left(\frac{1-\epsilon_n}{\epsilon_n}\right)$
- end.

Strong Classifier.

After every learning iteration, the score assigned by the strong classifier to a bag $I \in \mathcal{I}$ is:

$$S(I) = \sum_{n=1}^{N} \alpha_n \cdot h_n(I) .$$

Oza et al. 2001

Online Boosting

Initialization.

- Let $\mathcal{H} = \{h_1 ,..., h_N\}$ be a set of weak classifiers with Learning Principle L.
- Set $\lambda_n^w = \lambda_n^c = 0 \;\; \forall n \in \{1,...,N\}$.

Training.

At each iteration step t a novel sample I_t is presented to the system:

- Set the importance weight of the sample to $\lambda = 1$.
- For $n \in \{1,...,N\}$ do:
  1. update $h_n \leftarrow L(h_n, I_t, \lambda)$
  2. if h_n correctly classifies I_t:
     - $\lambda_n^c \leftarrow \lambda_n^c + \lambda\,; \;\; \epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}\,; \;\; \lambda \leftarrow \lambda \frac{1}{2(1-\epsilon_n)}$

     else:
     - $\lambda_n^w \leftarrow \lambda_n^w + \lambda\,; \;\; \epsilon_n = \frac{\lambda_n^w}{\lambda_n^c + \lambda_n^w}\,; \;\; \lambda \leftarrow \lambda \frac{1}{2\epsilon_n}$
  3. Define the relevance weight of the n-th weak learner as $\alpha_n = \log\left(\frac{1-\epsilon_n}{\epsilon_n}\right)$
- end.

Strong Classifier.

After every learning iteration, the score assigned by the strong classifier to a bag $I \in \mathcal{I}$ is:

$$S(I) = \sum_{n=1}^{N} \alpha_n \cdot h_n(I)\,.$$

Oza et al. 2001

Overview

- Why the hand?
- Multiple Instance Learning
- Online Boosting
- Online Multiple Instance Learning
- Experiments
- Conclusions

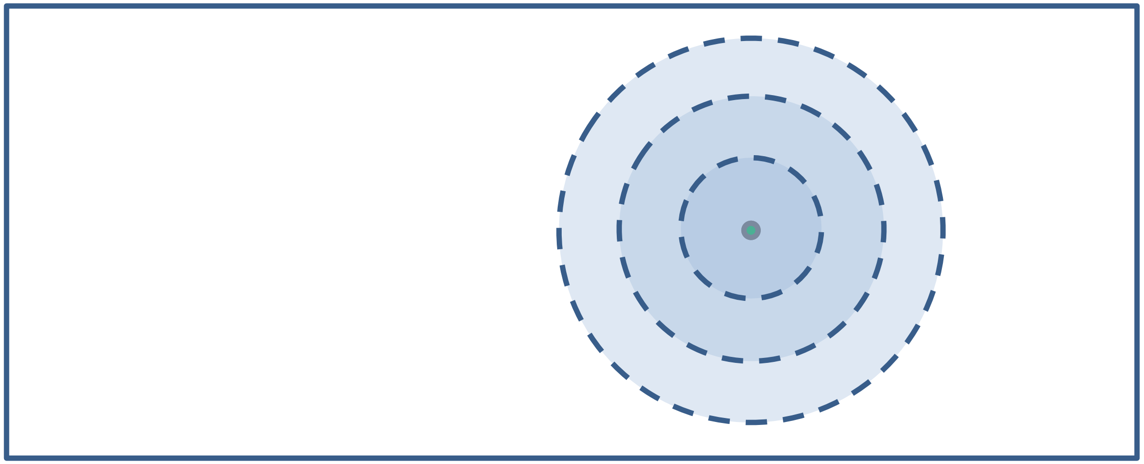
Weak Learners: online MIL Balls

Weak Learners: online MIL Balls

An online MIL weak learner is a MIL ball with variable radius.

Weak Learners: online MIL Balls

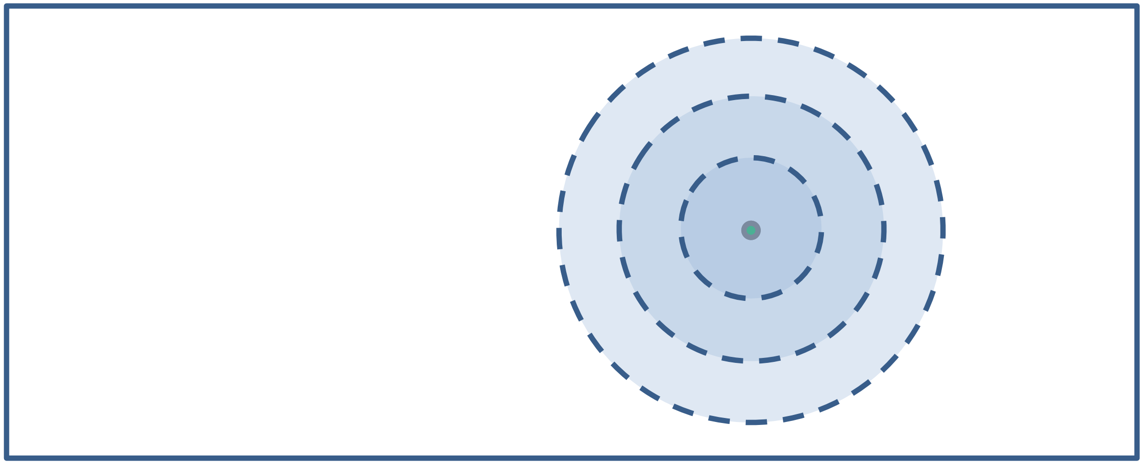
An online MIL weak learner is a MIL ball with variable radius.



Weak Learners: online MIL Balls

An online MIL weak learner is a MIL ball with variable radius.

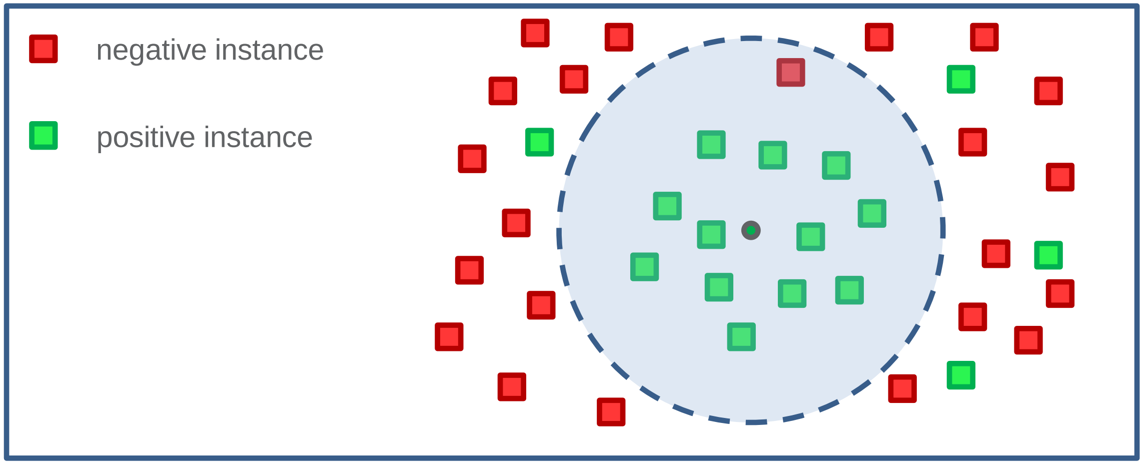
Learning principle: whenever a novel training sample is provided to the system, the radius of the weak learner is updated in order to keep its accuracy optimized.



Weak Learners: online MIL Balls

An online MIL weak learner is a MIL ball with variable radius.

Learning principle: whenever a novel training sample is provided to the system, the radius of the weak learner is updated in order to keep its accuracy optimized.



Feature Selection

Feature Selection

In an online framework, useful instances can arise from any bag in any moment.

Feature Selection

In an online framework, useful instances can arise from any bag in any moment.

Example: the object of interest rotates and shows previously hidden features.

Feature Selection

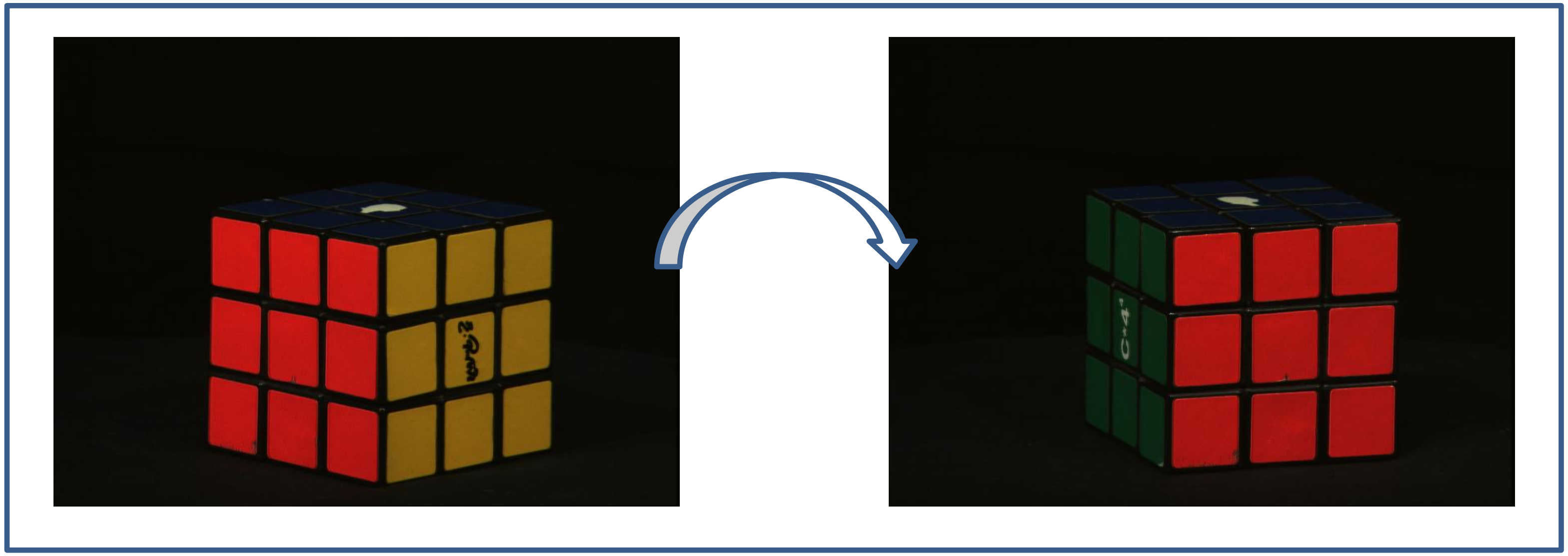
In an online framework, useful instances can arise from any bag in any moment.

Example: the object of interest rotates and shows previously hidden features.

Feature Selection

In an online framework, useful instances can arise from any bag in any moment.

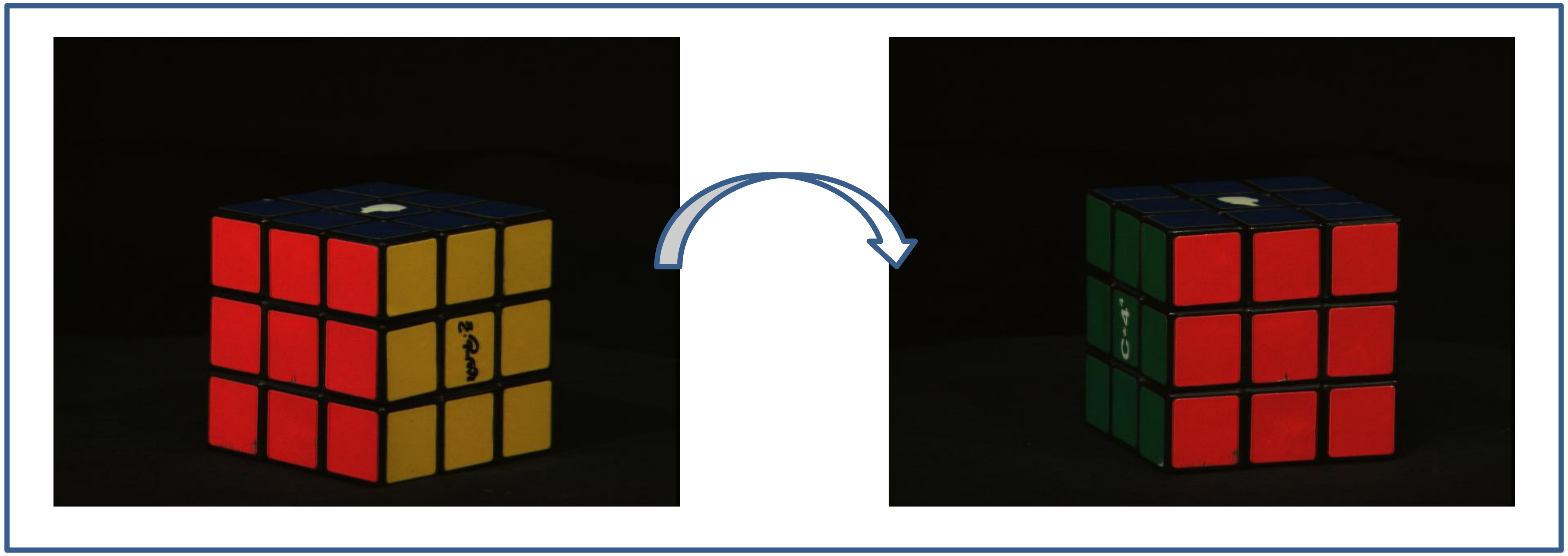
Example: the object of interest rotates and shows previously hidden features.



Feature Selection

In an online framework, useful instances can arise from any bag in any moment.

Example: the object of interest rotates and shows previously hidden features.



On the other hand, the online boosting algorithm requires all the weak learners to be determined a priori. It is not possible to extract new ones from the data.

Weak Learners: Selectors

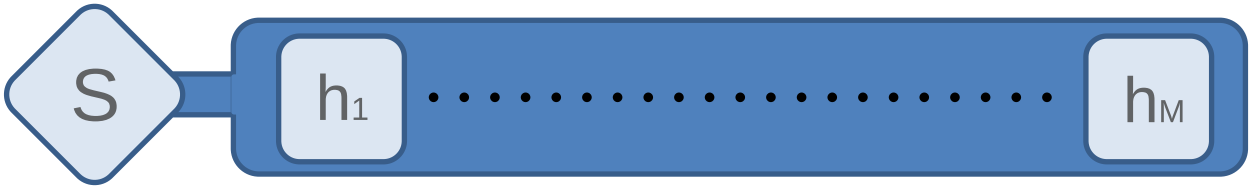
Weak Learners: Selectors

A selector is a meta-weak learner associated to a fixed pool of other weak classifiers.

Grabner et al. 2006

Weak Learners: Selectors

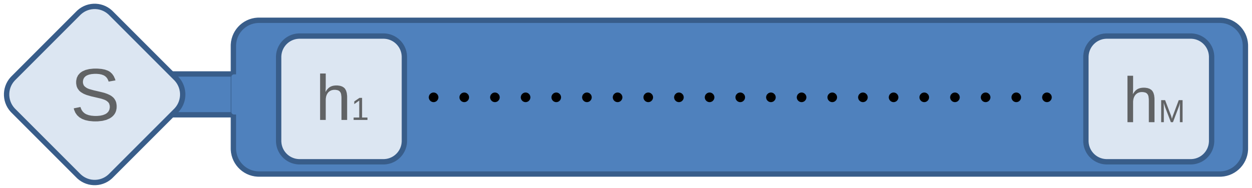
A selector is a meta-weak learner associated to a fixed pool of other weak classifiers.



Grabner et al. 2006

Weak Learners: Selectors

A selector is a meta-weak learner associated to a fixed pool of other weak classifiers.

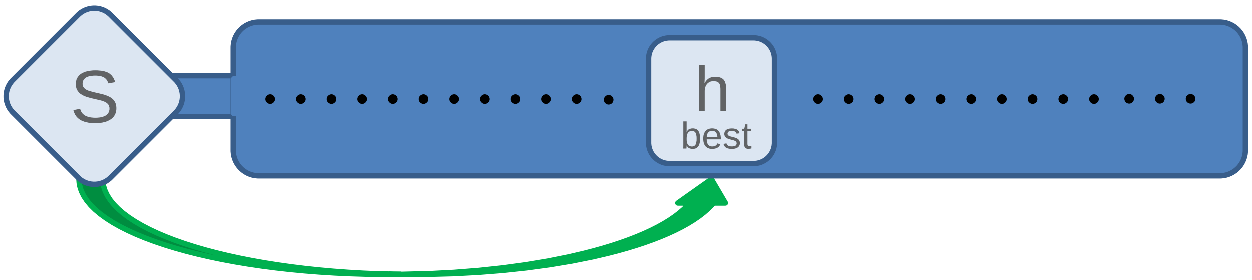


Decision Criteria: The selector classify a bag, according to the weak classifier in its pool that has exhibited the best accuracy so far.

Grabner et al. 2006

Weak Learners: Selectors

A selector is a meta-weak learner associated to a fixed pool of other weak classifiers.



Decision Criteria: The selector classify a bag, according to the weak classifier in its pool that has exhibited the best accuracy so far.

Grabner et al. 2006

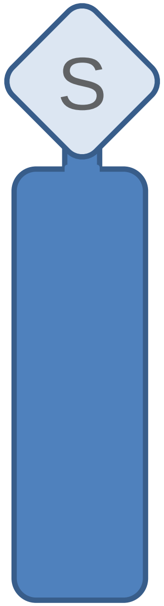
Weak Learners: Selectors

Learning principle:

Grabner et al. 2006

Weak Learners: Selectors

Learning principle: When a novel training sample is presented to the selector, each of the weak learners in its pool is trained accordingly.

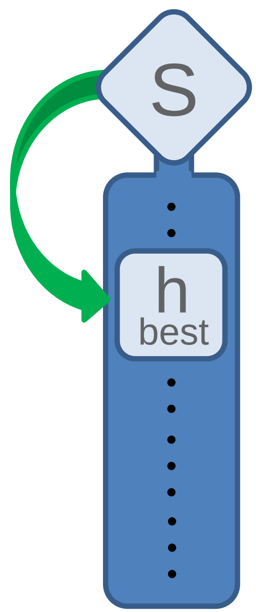


Grabner et al. 2006

Weak Learners: Selectors

Learning principle: When a novel training sample is presented to the selector, each of the weak learners in its pool is trained accordingly.

The weak learner with lowest error rate is selected.



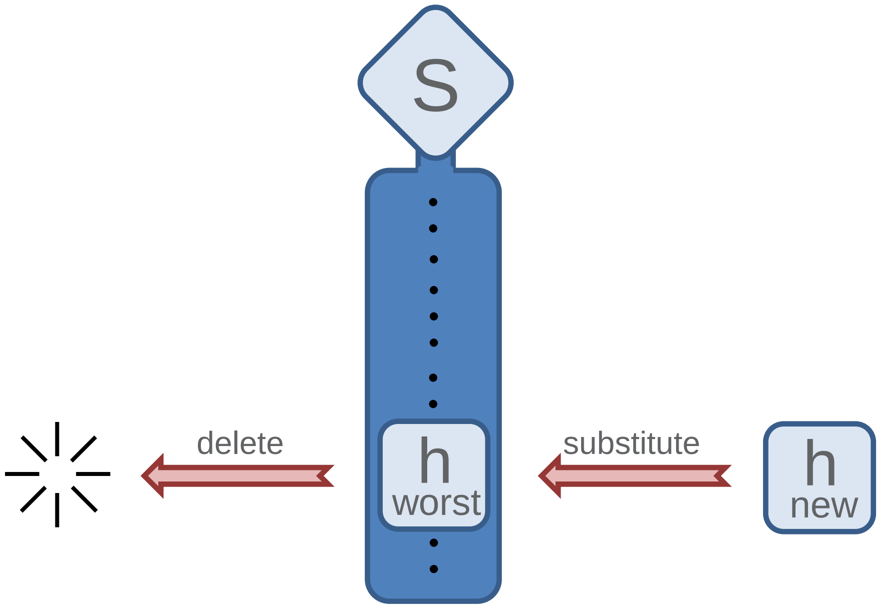
Grabner et al. 2006

Weak Learners: Selectors

Learning principle: When a novel training sample is presented to the selector, each of the weak learners in its pool is trained accordingly.

The weak learner with lowest error rate is selected.

The weak learner with worst classification performance is substituted with a new one.



Grabner et al. 2006

Main Framework

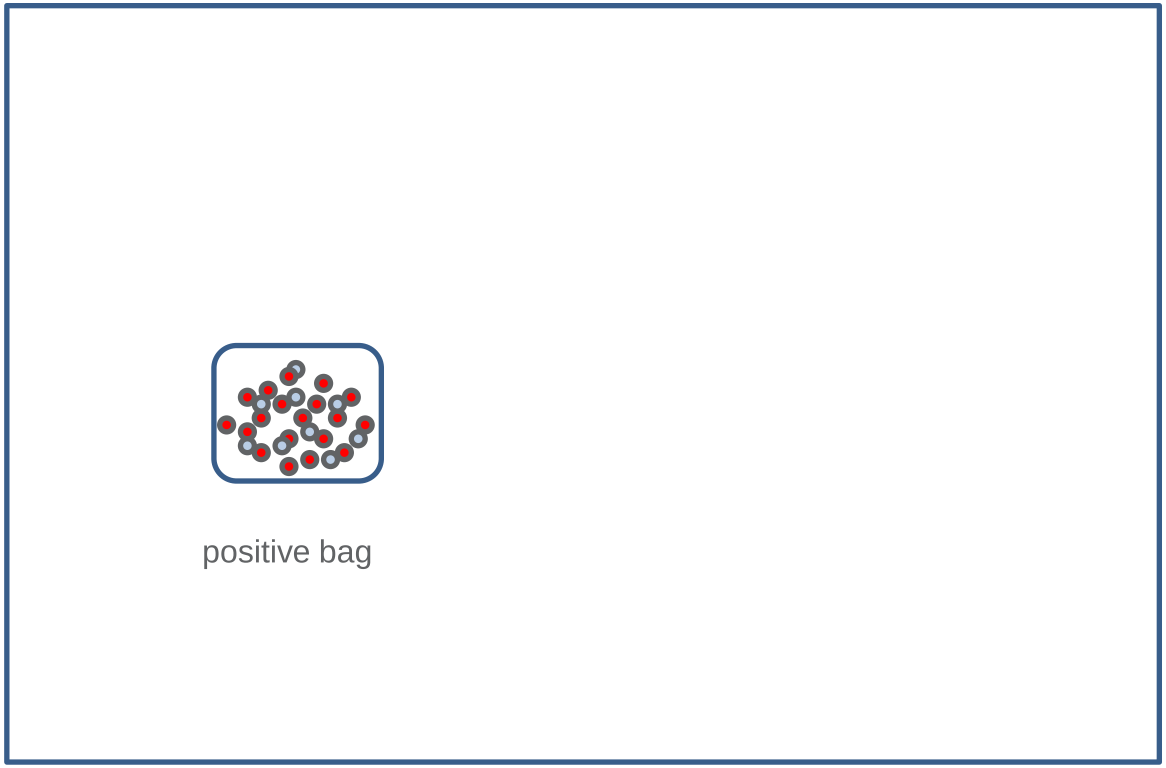
- New weak learners extraction.
- Online boosting of selectors.

Main Framework

- New weak learners extraction.
- Online boosting of selectors.

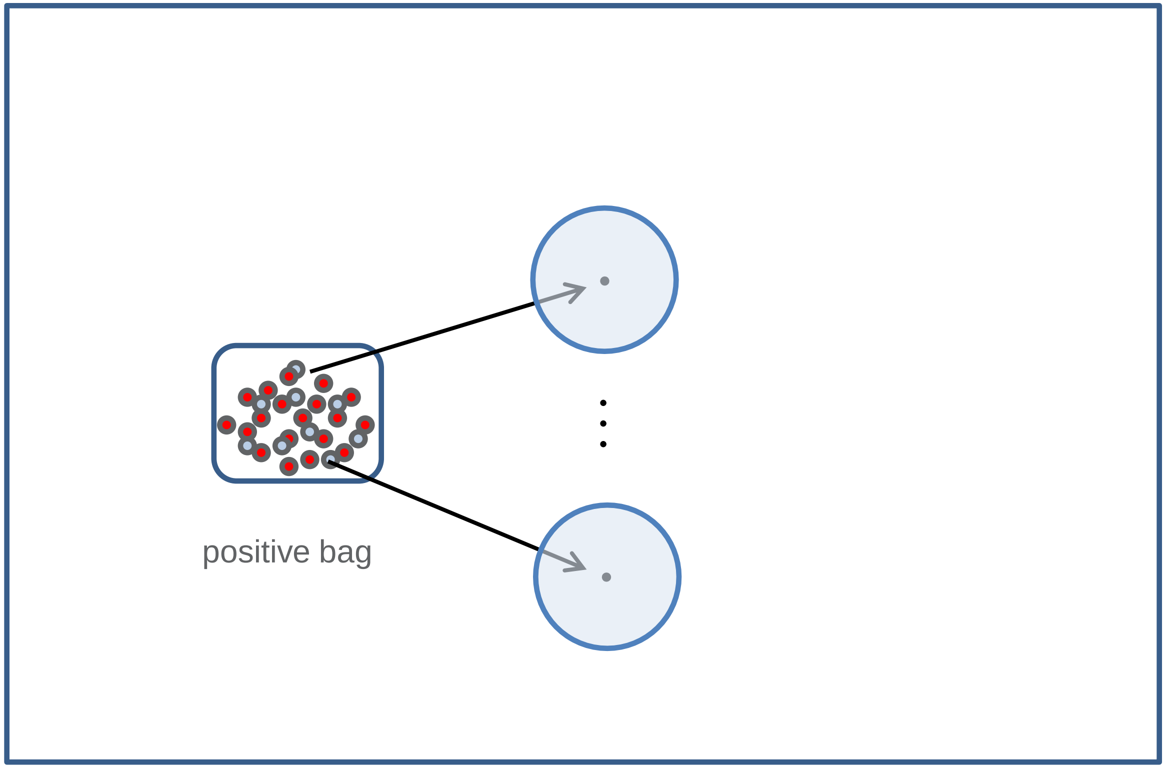
Main Framework

- New weak learners extraction.
- Online boosting of selectors.



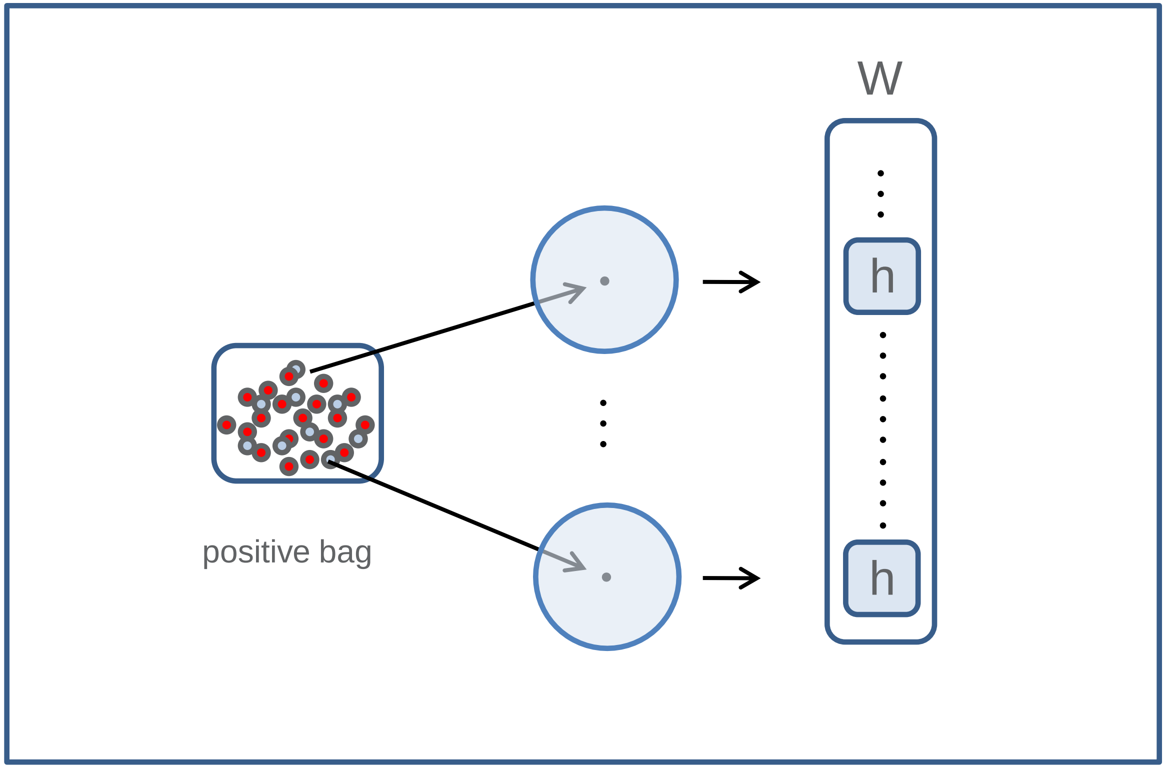
Main Framework

- New weak learners extraction.
- Online boosting of selectors.



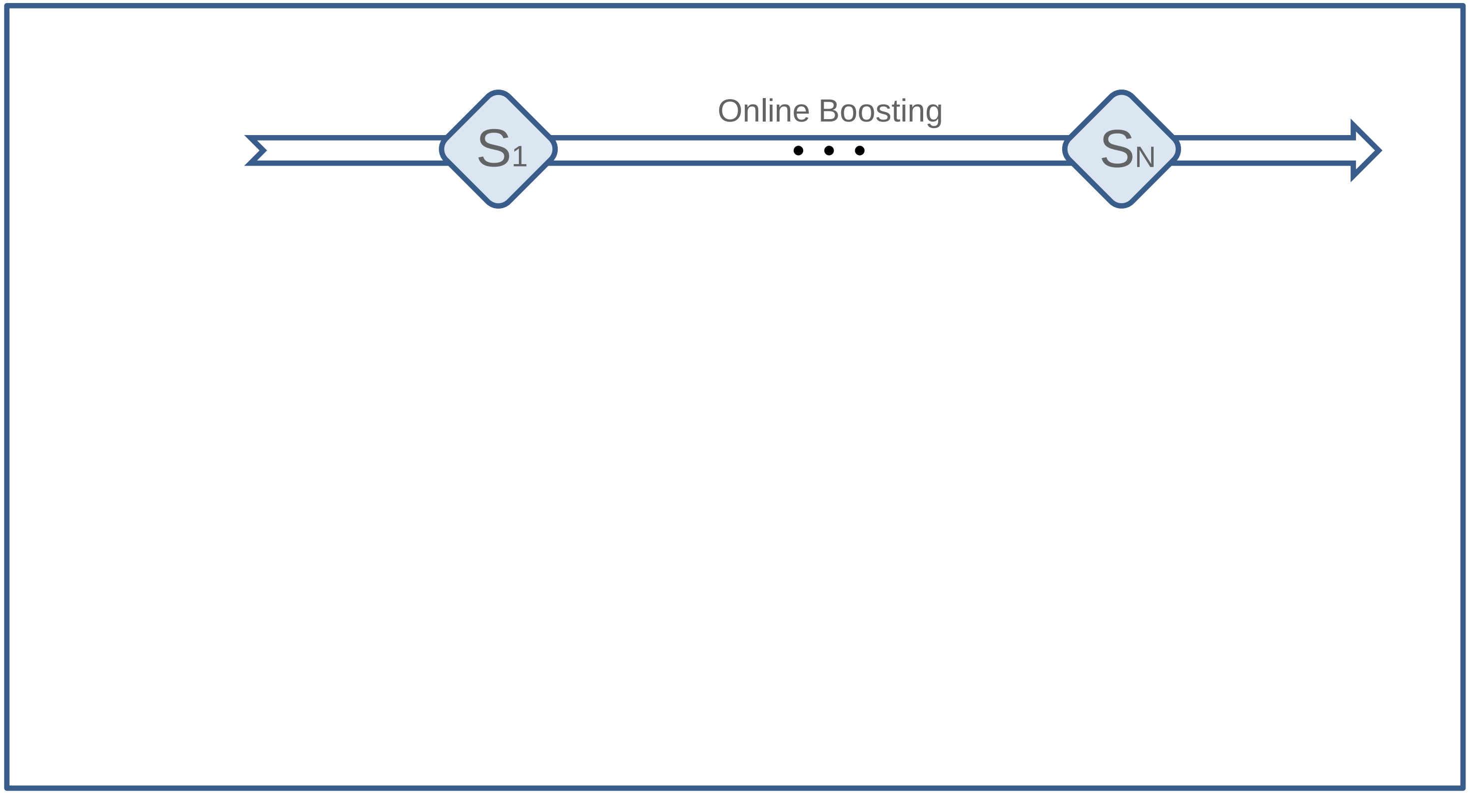
Main Framework

- New weak learners extraction.
- Online boosting of selectors.



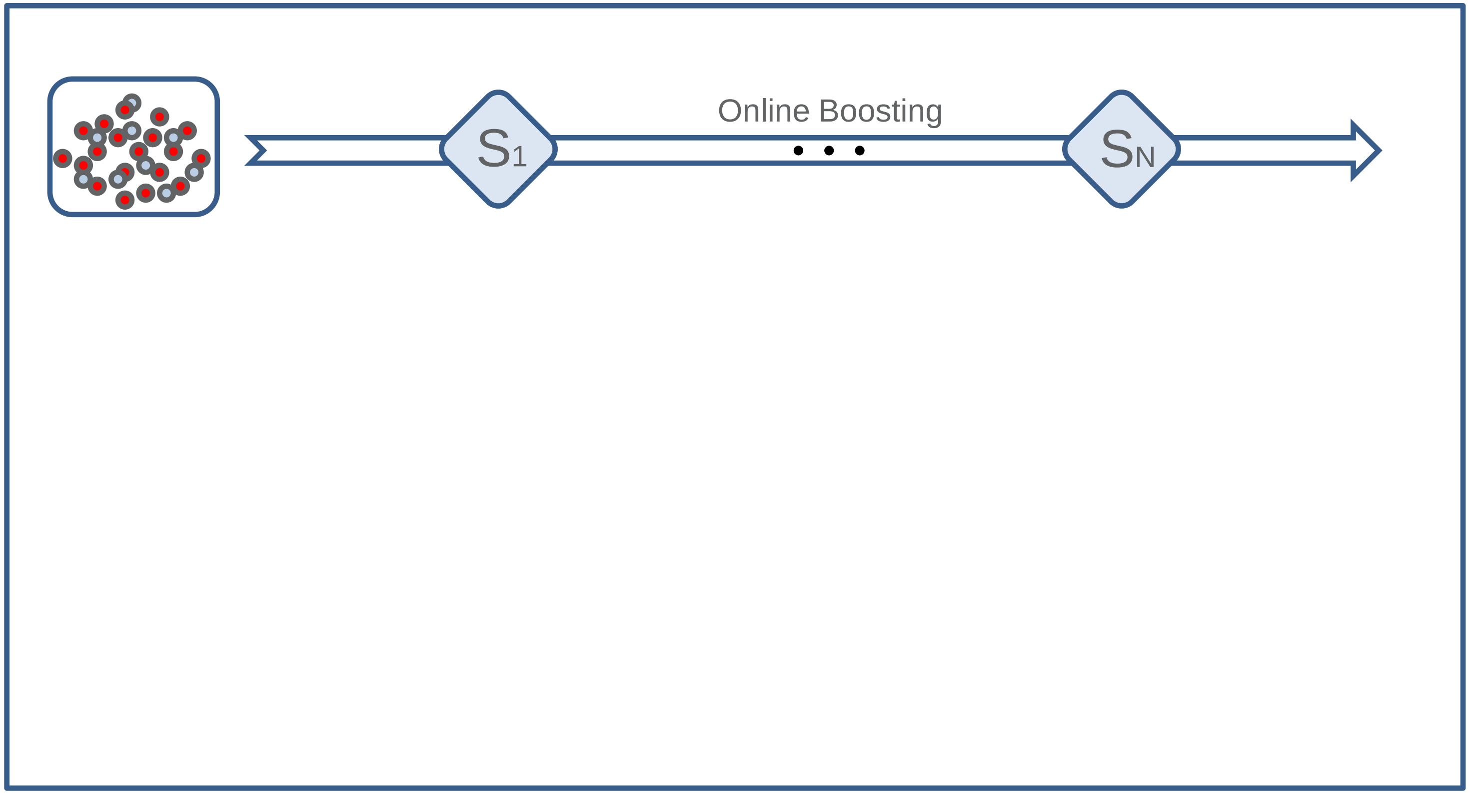
Main Framework

- New weak learners extraction.
- Online boosting of selectors.



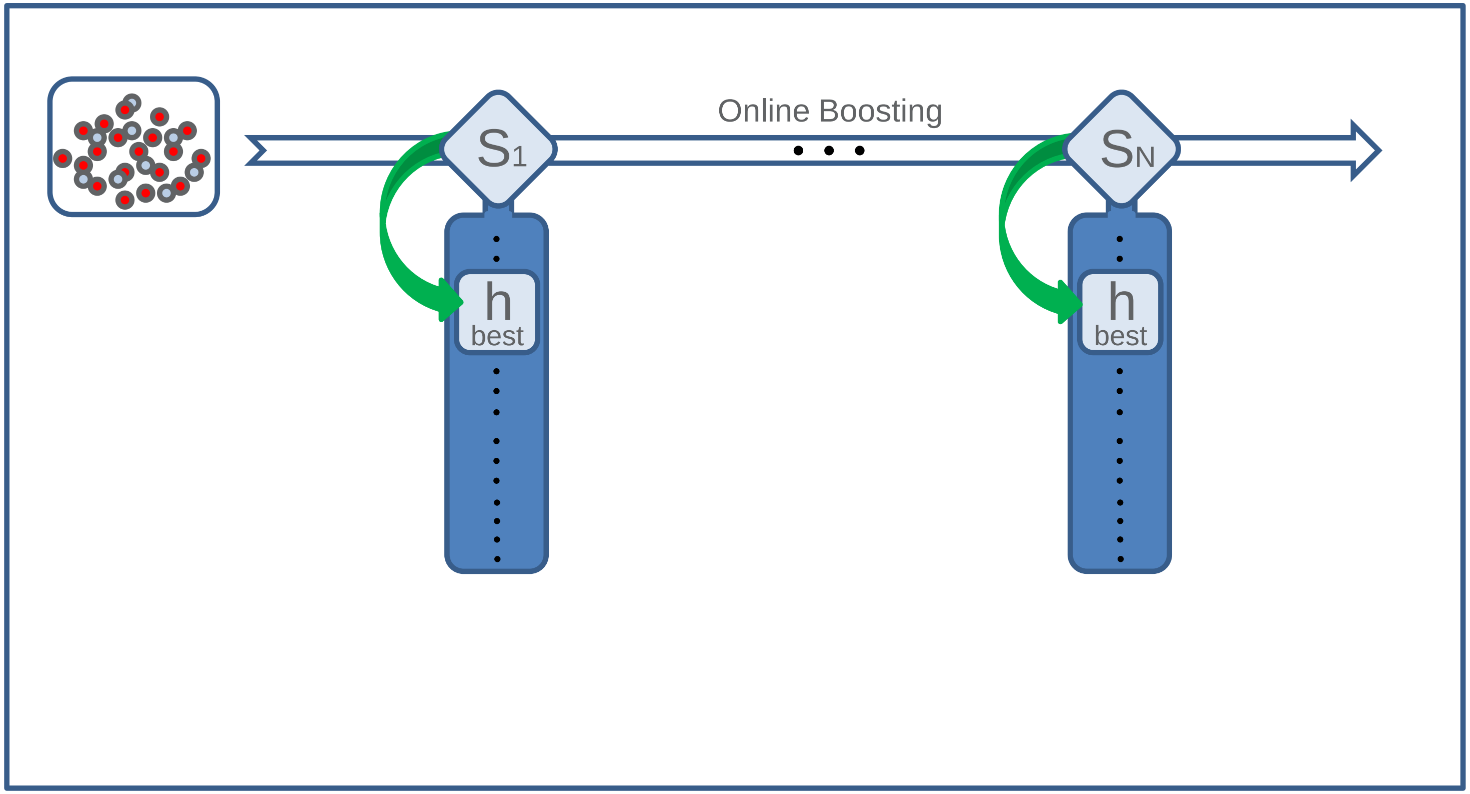
Main Framework

- New weak learners extraction.
- Online boosting of selectors.



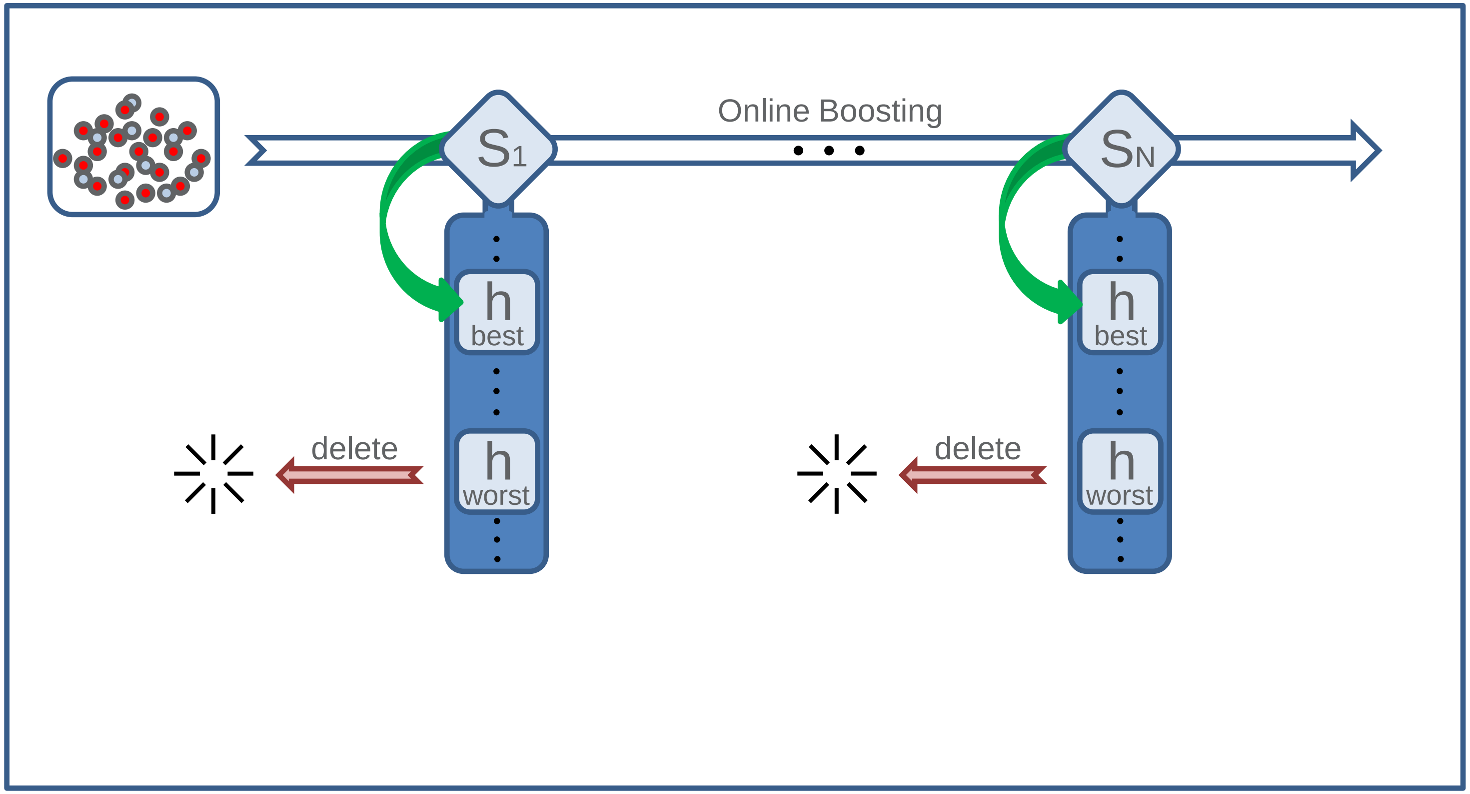
Main Framework

- New weak learners extraction.
- Online boosting of selectors.



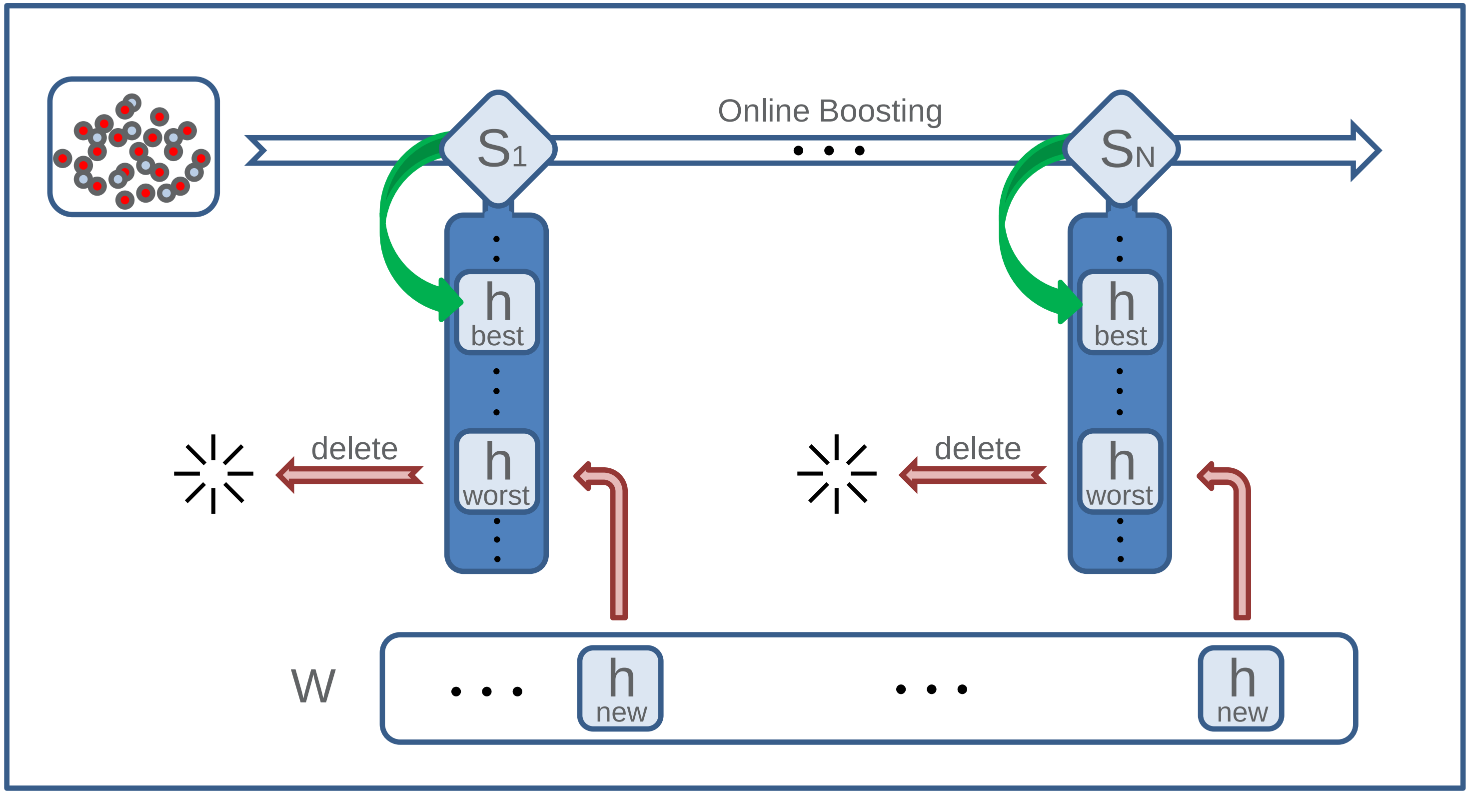
Main Framework

- New weak learners extraction.
- Online boosting of selectors.



Main Framework

- New weak learners extraction.
- Online boosting of selectors.



Overview

- Why the hand?
- Multiple Instance Learning
- Online Boosting
- Online Multiple Instance Learning
- Experiments
- Conclusions

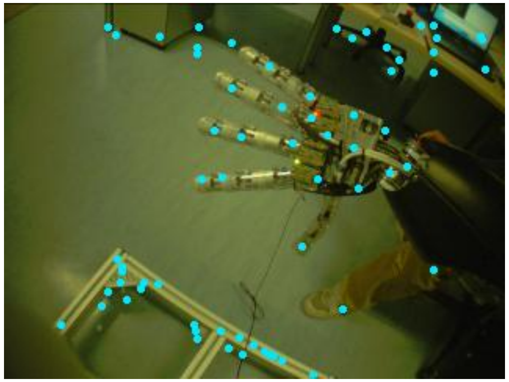
Hand Detection

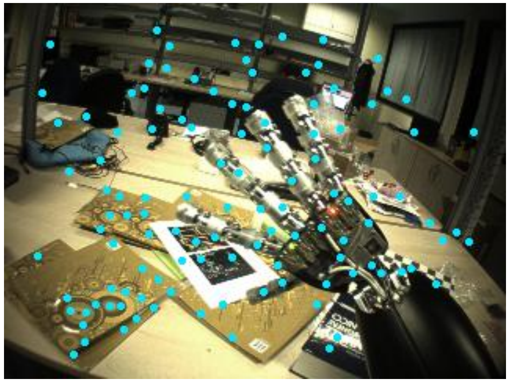
Different experimental conditions:

Hand Detection

Different experimental conditions:

- **Background:** Uniform Vs Cluttered.

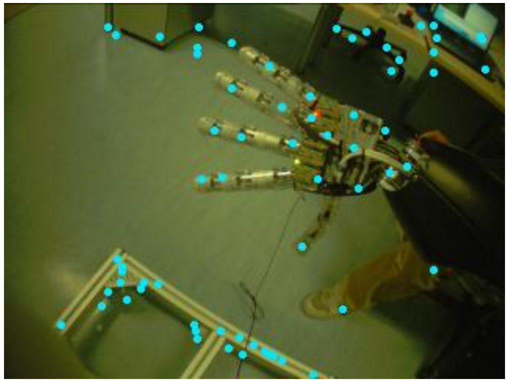


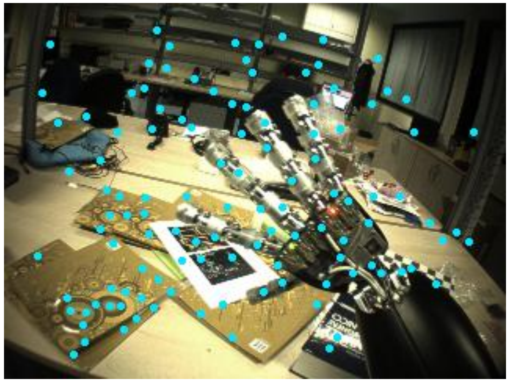


Hand Detection

Different experimental conditions:

• **Background:** Uniform Vs Cluttered.



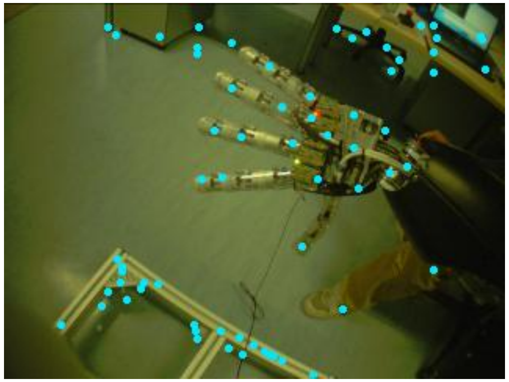


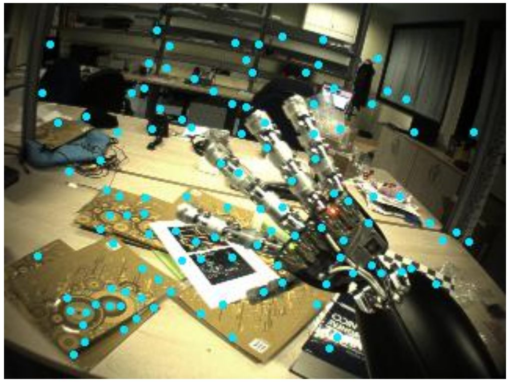
• **Labeling:** Manual Vs Automatic.

Hand Detection

Different experimental conditions:

- **Background:** Uniform Vs Cluttered.

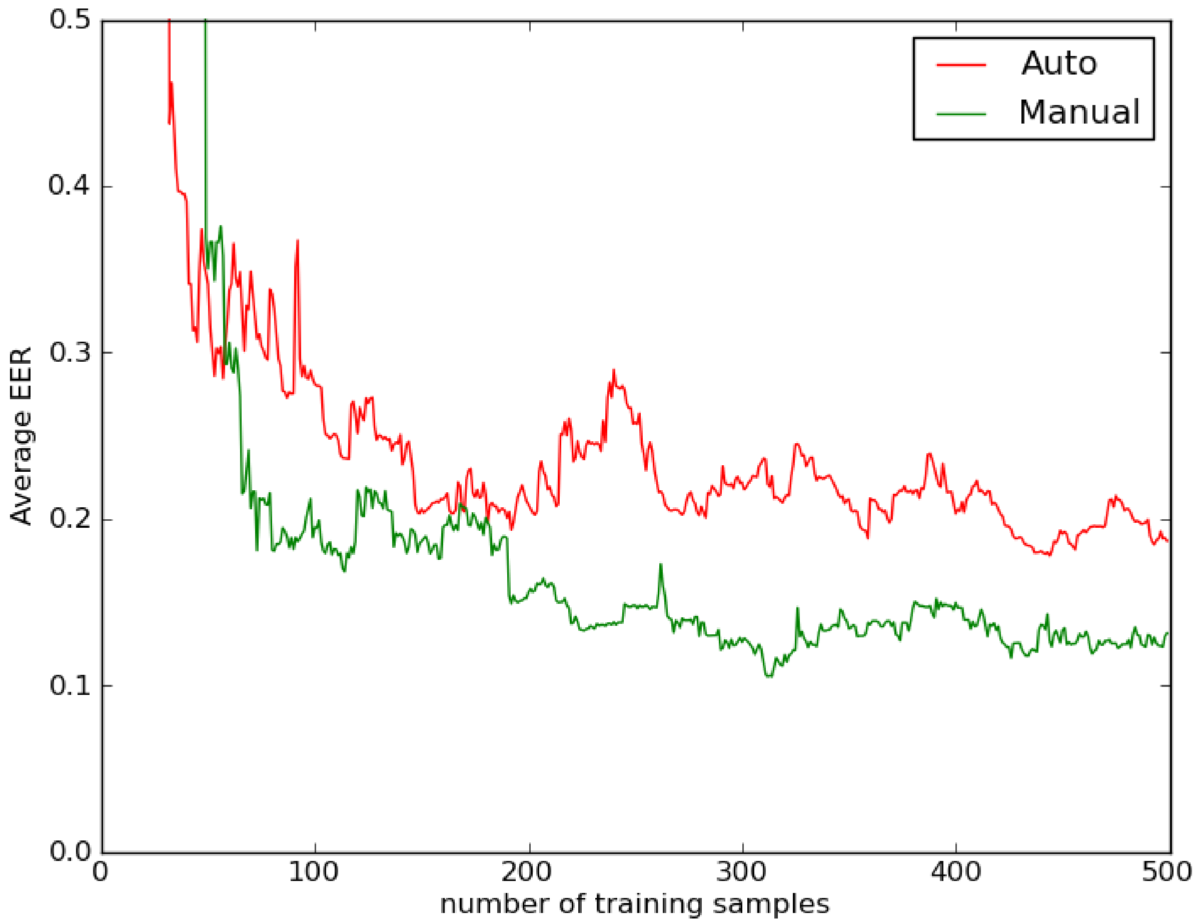




- **Labeling:** Manual Vs Automatic.

- **Order:** Natural Vs Shuffled.

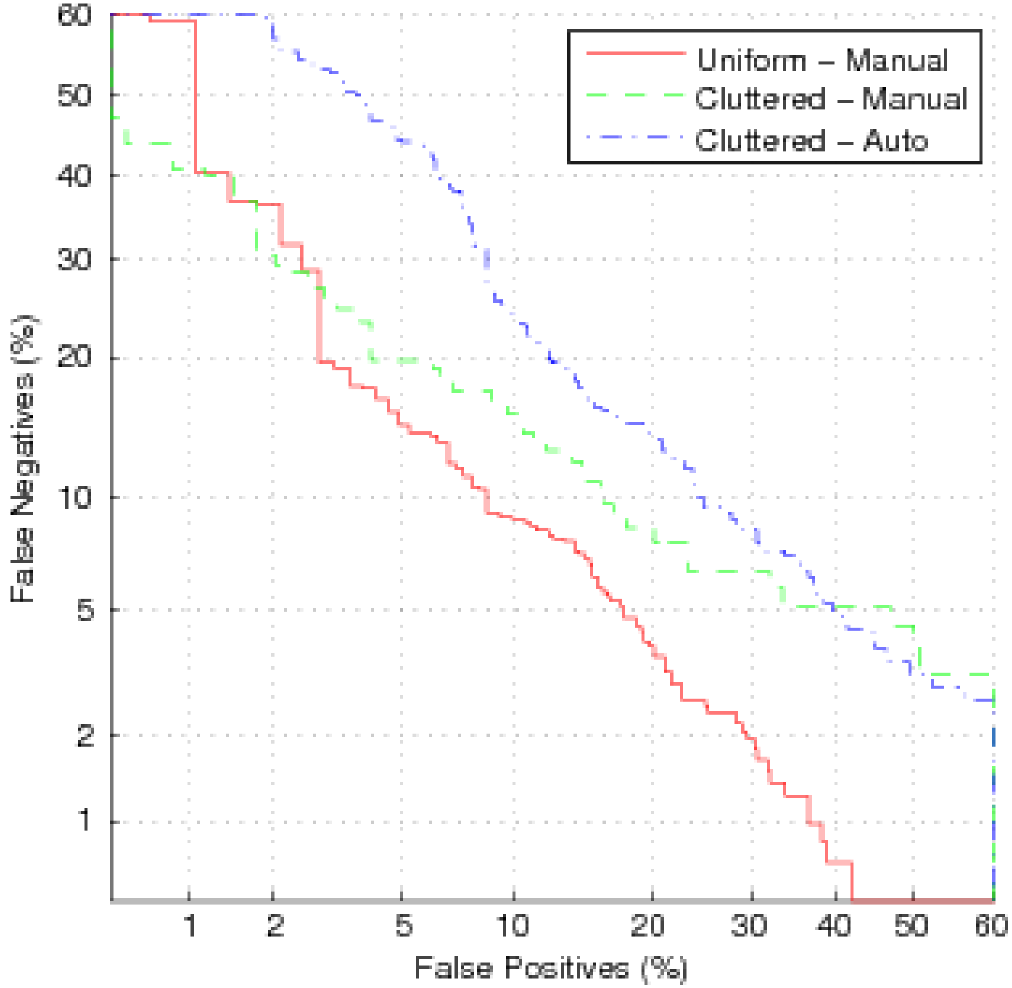
Hand Detection: learning over time



Hand Detection: statistics

Dataset	Labelling	Order	EER Online	EER Offline
uniform	Manual	Natural	$(9.0 \pm 0.7)\%$	7.3%
uniform	Manual	Shuffled	$(7.4 \pm 2.5)\%$	7.3%
cluttered	Manual	Natural	$(13.0 \pm 1.9)\%$	7.5%
cluttered	Manual	Shuffled	$(12 \pm 1)\%$	7.5%
cluttered	Auto	Natural	$(18 \pm 2)\%$	9.9%
cluttered	Auto	Shuffled	$(19 \pm 5)\%$	9.9%

Hand Detection: ROC curves



Overview

- Why the hand?
- Multiple Instance Learning
- Online Boosting
- Online Multiple Instance Learning
- Experiments
- Conclusions

Conclusions

- Multiple Instance Learning needs only weak supervision over data in order to be trained (e.g. positive or negative label over entire samples).

- The MIL paradigm allows to deal with **inaccurate** and possibly **noisy** training data (e.g. coarse labeling of images).

- In our framework learning is performed online in order to adapt to potential changes in the scene (e.g. illumination or orientation).

- Experiments were conducted on a real problem and under quite hard conditions (e.g. cluttered background). Nevertheless online performances remained comparable to those of batch algorithms.

Future work: localization

Selectors have originally been conceived to perform **feature selection**.

Future work: localization

Selectors have originally been conceived to perform **feature selection**.

Feature selection could be exploited to perform hand localization:

Future work: localization

Selectors have originally been conceived to perform **feature selection**.

Feature selection could be exploited to perform hand localization:

The positive classifying weak learners within the selector pools can be plotted on the current image.

Future work: localization

Selectors have originally been conceived to perform **feature selection**.

Feature selection could be exploited to perform hand localization:

The positive classifying weak learners within the selector pools can be plotted on the current image.

In theory they should all be on the object.

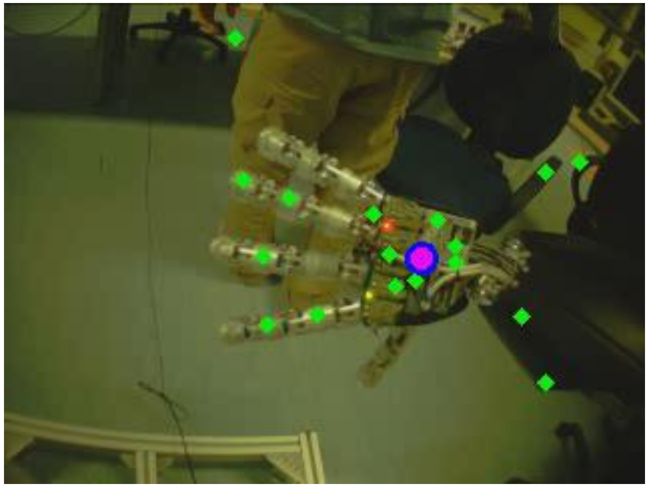
Future work: localization

Selectors have originally been conceived to perform **feature selection**.

Feature selection could be exploited to perform hand localization:

The positive classifying weak learners within the selector pools can be plotted on the current image.

In theory they should all be on the object.



Future work: localization

Thanks:

- Fabrizio Smeraldi
- Lorenzo Natale
- Giorgio Metta

Thank you!